



# GLAST

*The Gamma Ray Large Area Space Telescope*

<b>GLAST LAT ACD ELECTRONICS SUBSYSTEM TECHNICAL DOCUMENT</b>	Document # <b>ACD-PROC-000062</b>	Date Effective 10-17-2003
	Prepared by: J. L. Odom and D. A. Sheppard	Supersedes None
Document Title <b>GLAST LAT ACD GARC Comprehensive Performance Test</b>		

**Gamma-ray Large Area Space Telescope (GLAST)  
Large Area Telescope (LAT)  
Anti-Coincidence Detector (ACD)**

**GARC ASIC (GLAST ACD Readout Controller)**



**Comprehensive Performance Test**

**Rev (–) October 17, 2003**

## Document Approval

Prepared by:

\_\_\_\_\_  
James Odom                      Date  
ACD Electronics

Prepared by:

\_\_\_\_\_  
Dave Sheppard                      Date  
ACD Electronics

Approved by:

\_\_\_\_\_  
Glenn Unger                      Date  
ACD Electronics System Lead

Approved by:

\_\_\_\_\_  
George Shibley                      Date  
ACD Systems Engineer

Approved by:

\_\_\_\_\_  
Tom Johnson                      Date  
ACD Project Manager

Approved by:

\_\_\_\_\_  
Dave Thompson                      Date  
ACD Subsystems Manager

Approved by:

\_\_\_\_\_  
Eileen Fowler                      Date  
Quality Assurance

Approved by:

\_\_\_\_\_  
Bob Hartman                      Date  
ACD Scientist

## GARC Test Table of Contents

1.0	Description of the GARC ASIC .....	6
2.0	Preconditions to and Preparations for Starting this Test .....	6
2.1	GARC Chip Identification .....	6
2.2	Test Equipment Utilized .....	7
3.0	GARC Command Mnemonics and Functions .....	7
4.0	Testing the GARC Digital Logic .....	10
4.1	Measurement of the GARC Bias Resistor Voltages .....	11
4.2	Initial Reset Test .....	11
5.0	GARC Power Measurements .....	15
5.1	Measurement at the Nominal Power Supply Voltage .....	15
5.2	Measurement at the Minimum Power Supply Voltage .....	15
5.3	Measurement at the Maximum Power Supply Voltage .....	16
6.0	GARC Register Read/Write and PHA Readout Tests .....	17
6.1	Veto_Delay Register Test .....	18
6.2	HVBS Level Register Test .....	18
6.3	SAA Level Register Test .....	18
6.4	Hold Delay Register Test .....	18
6.5	Veto Width Register Test .....	19
6.6	HitMap Width Register Test .....	19
6.7	HitMap Deadtime Register Test .....	19
6.8	HitMap Delay Register Test .....	20
6.9	PHA En0 Register Test .....	20
6.10	Veto En0 Register Test .....	20
6.11	PHA En1 Register Test .....	20
6.12	Veto En1 Register Test .....	21
6.13	MaxPHA Register Test .....	21
6.14	GARC Mode Register Test .....	21
6.15	PHA Threshold Channel 00 Register Test .....	22
6.16	PHA Threshold Channel 01 Register Test .....	22
6.17	PHA Threshold Channel 02 Register Test .....	22
6.18	PHA Threshold Channel 03 Register Test .....	23
6.19	PHA Threshold Channel 04 Register Test .....	23
6.20	PHA Threshold Channel 05 Register Test .....	23
6.21	PHA Threshold Channel 06 Register Test .....	24
6.22	PHA Threshold Channel 07 Register Test .....	24
6.23	PHA Threshold Channel 08 Register Test .....	24
6.24	PHA Threshold Channel 09 Register Test .....	24
6.25	PHA Threshold Channel 10 Register Test .....	25
6.26	PHA Threshold Channel 11 Register Test .....	25
6.27	PHA Threshold Channel 12 Register Test .....	25
6.28	PHA Threshold Channel 13 Register Test .....	26
6.29	PHA Threshold Channel 14 Register Test .....	26
6.30	PHA Threshold Channel 15 Register Test .....	26
6.31	PHA Threshold Channel 16 Register Test .....	26
6.32	PHA Threshold Channel 17 Register Test .....	27
6.33	ADC TACQ Register Test .....	27
6.34	GAFE ASICs Mode Register Test .....	27
6.35	GAFE ASIC VETO DAC Register Test .....	28
6.36	GAFE ASIC VETO VERNIER Register Test .....	28

6.37	GAFE ASIC HLD Register Test.....	28
6.38	GAFE ASIC BIAS Register Test.....	29
6.39	GAFE ASIC TCI Register Test.....	29
6.40	Test of the GARC Parity Error Detection and Error Generation .....	30
6.41	Test of the GARC Diagnostic Status Register .....	30
6.42	Command Counter Test .....	31
6.43	Test of the Look-At-Me Circuitry .....	31
7.0	Test of the GARC PHA Functions .....	32
7.1	Maximum PHA Return Test .....	32
7.2	PHA Enable/Disable Test .....	32
7.3	PHA Threshold Verification Test .....	34
8.0	Test of the MAX5121 DAC and Associated High Voltage Supply Control Functions .....	36
8.1	Capture of the DAC Control Signals.....	36
8.2	Test of the GARC SAA/HV Normal Modes.....	36
8.3	Test of the HVBS Triple Modular Redundancy Circuitry .....	37
9.0	GARC Test Pin Mux Verification .....	39
10.0	Test of the Hold Delay Operation.....	39
11.0	Test of the AEM VETO Signal Functionality .....	43
11.1	Veto Delay Test.....	43
11.2	Veto Width Test .....	45
11.3	Veto Enable and Disable Test .....	45
11.4	Discriminator Input Minimum Width Test .....	47
11.5	Discriminator Input Extended Width Test .....	49
11.6	Discriminator Input Double Pulse Test.....	50
12.0	Test of the HitMap Functionality .....	51
12.1	HitMap Width Test .....	51
12.2	HitMap Delay Test.....	52
12.3	HitMap Deadtime Stretch Test.....	53
12.4	HitMap Minimum Width Test .....	54
12.5	HitMap Extended Pulse Test.....	54
12.6	HitMap Double Pulse Test.....	55
13.0	Strobe Test.....	55
14.0	Capture of the ADC Control Signals.....	56
15.0	ADC TACQ Test.....	57
16.0	Test of the GARC LVDS Circuitry Driver Currents .....	58
17.0	Multiple System Clock Speed Test and Power Supply Rail Tests – 3.0V to 3.6V .....	61
	Appendix 1: GARC Documentation.....	63
	Appendix 2: GARC Configuration Command Format.....	63
	Appendix 3: GARC Event Data Format:.....	64
	Appendix 4: GARC Configuration Data Readback Format: .....	64
	Appendix 5: GARC Pin List.....	64
	Appendix 6: Test Results Record .....	69

### Document Change Log

Revision	Date	Change Description	Prepared by
Initial Draft	May, 2003	Initial Draft	Dave Sheppard
Updated	September 22, 2003	Updates from first GARC testing	Jim Odom and Dave Sheppard
-	October 17, 2003	Final Version	Jim Odom and Dave Sheppard

## 1.0 Description of the GARC ASIC

GARC is the acronym for the Gamma-Ray Large Area Space Telescope (GLAST) Anti-Coincidence Detector (ACD) Readout Controller Application Specific Integrated Circuit (ASIC). The GARC is the main logical interface for the ACD to the LAT instrument electronics. GARC provides command and data return functions for electronics boards associated with the ACD. There are 12 GARCs in the flight system, one for each of the event electronics cards. The GARC is a +3.3V single supply device and communicates to the LAT digitally via LVDS interfaces. The GARC is packaged in a 208 pin plastic quad flatpack.

This document describes the functions and test plan of the GARC ASIC. The GARC is designed to meet the requirements of the ACD Level IV Electronics Requirements, LAT-SS-00352, and the ACD-LAT Interface Control Document, LAT-SS-00-363.

The design was done utilizing Verilog as the description language, Exemplar Leonardo Spectrum as the synthesis tool targeting the Tanner Agilent 0.5  $\mu\text{m}$  standard cell library, and Tanner L-Edit as the automated place and route layout tool. Core verification was performed via the Tanner LVS tool.

## 2.0 Preconditions to and Preparations for Starting this Test

This is the nominal functional test used to verify that the GARC meets the specified requirements. Prior to starting this test, the power supply to the GARC test board should be verified to be at the correct voltage, which is nominally +3.3V. Variations of this test may include voltages in the +3.0V to +3.6V range, temperatures in the range of -20C to +60C, and different clock frequencies. Nominally, the ACD clock frequency is to be 20 MHz.

Notify QA 24 hours prior to the start of this test. QA will verify test equipment set up and calibration, review documentation and decide if their presence is required during the testing. Indicate on the test record whether QA was present for the testing.

Additionally, prior to starting the functional test, the proper biasing of the GARC circuitry is to be verified. The following table details the expected biases.

GARC Bias Signal	GARC Pin	Nominal Bias Resistor Required	GARC Test Board Res	Function of Bias Resistor
HLD_WOR_BIAS	104	7.50 k $\Omega$ to DGND	R19	HLD Wired-OR Rcvr Bias
BIAS_RCVR	156	82.5 k $\Omega$ to VDD	R16	LVDS Rcvr Bias Adjust
BIAS_DRV_H	160	4.53 k $\Omega$ to DGND	R15	LVDS Driver Bias Adjust
BIAS_DRV_L	169	15.0 k $\Omega$ to DGND	R13	LLDRV Driver Bias Adjust
LVDS_PRESETADJ	184	3.92 k $\Omega$ to VDD	R20	LVDS Preset Adjust

An additional variation possible to this test is to individually vary the bias resistors by some nominal amount (e.g., 10%) and utilize this functional test to determine variations in performance, if any.

All measurements and test results will be recorded in the "Test Results Record" Appendix 6. A copy of the entire this entire document is not required each time the test is performed. The appendix will serve as the official record each time this test is performed.

## 2.1 GARC Chip Identification

The test conductor for this test is: \_\_\_\_\_

Date of this test is: \_\_\_\_\_

The identification/Serial Number listed on the GARC ASIC is: \_\_\_\_\_

The serial number of the AEM Simulator board is: \_\_\_\_\_

The serial number of the GARC Test board is: \_\_\_\_\_

The serial number of the GAFE Simulator board is: \_\_\_\_\_

## 2.2 Test Equipment Utilized

Prior to starting this test, the Test Conductor shall record the test and measurement hardware used in the performance of this test. Note that all multimeters used in testing of the FREE circuit card assembly shall be of low current output design, such as the Fluke 70 series, HP3400 series or similar. High current output multimeters are not compatible with the FREE circuitry.

Instrument Type	Manufacturer & Model Number	NASA ID Number	Calibration Due Date
Power Supply +3.3V			
Power Supply +5.0V			
Multimeter 1			
Multimeter 2			
Pulse Generator			
Oscilloscope			

## 3.0 GARC Command Mnemonics and Functions

The following table represents each of the available GARC commands. Additionally, all GAFE commands are passed through the GARC. These command patterns are detailed in the document discussing the GAFE logic. There are two types of GARC commands – trigger commands (4 bits in length) and configuration commands (34 bits in length). The AEM-ACD ICD contains the authoritative formats for all command types, but they are repeated in the appendices of this document for convenience. The table below defines the command mnemonics that will be used in this test.

Note that a GAFE will process a write command either for the address hard-wired to the chip address pins or to an address of 'h1F, the GAFE broadcast address. A GAFE will process a read command only for an address identical to the hard-wired address. It is an operational constraint that, for any given ACD circuit board, each GAFE must have a unique address.

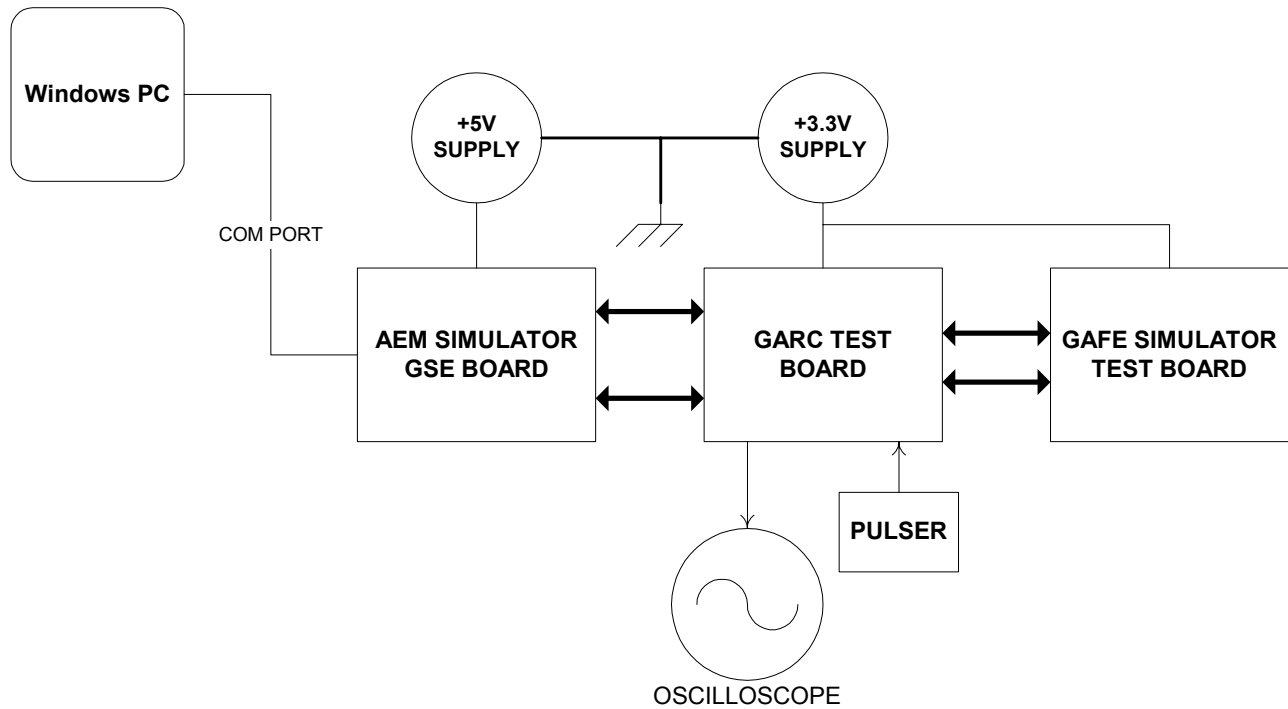
GARC Cmd No.	ACD Command Mnemonic	Rd/Wr Status	Select GARC=0 GAFE=1	Function Block	Register Number	No. of Data Bits	Command Function
1	Trigger_ZS	N/A	N/A	N/A	N/A	N/A	ACD Trigger, Zero-Suppression Enable
2	Trigger_NOZS	N/A	N/A	N/A	N/A	N/A	ACD Trigger, No Zero-Suppression
3	GARC_Reset	W	0	0	1	0	Generates reset for GARC and GAFE registers
4	Veto_Delay_Wr	W	0	0	2	5	Sets Delay from Disc_In to VETO Out
5	Veto_Delay_Rd	R	0	0	2	5	Reads contents of Veto_Delay register
6	GARC_Cal_Strobe	W	0	0	3	0	Sends Calibration Strobe signal to all GAFEs
7	HVBS_Level_Wr	W	0	0	8	12	Sets GARC register value from which HVBS may be commanded in the science mode
8	HVBS_Level_Rd	R	0	0	8	12	Reads contents of HVBS Level register
9	SAA_Level_Wr	W	0	0	9	12	Sets GARC register value from which HVBS may be commanded when in the SAA
10	SAA_Level_Rd	R	0	0	9	12	Reads contents of SAA Level register
11	Use_HV_Normal	W	0	0	10	0	Sends 12 bit value in HVBS Level register to the MAX5121 DAC
12	DAC_HVReg_Rd	R	0	0	10	0	Reads MAX5121 DAC Config Register
13	Use_SAA_Normal	W	0	0	11	0	Sends 12 bit value in SAA Level register to the MAX5121 DAC
14	DAC_SAAREg_Rd	R	0	0	11	0	Reads MAX5121 DAC Config Register
15	Hold_Delay_Wr	W	0	0	12	7	Sets GARC Hold Delay
16	Hold_Delay_Rd	R	0	0	12	7	Reads back value of GARC Hold Delay register
17	Veto_Width_Wr	W	0	0	13	3	Sets width of the VETO signals
18	Veto_Width_Rd	R	0	0	13	3	Reads back width of the VETO width register
19	HitMap_Width_Wr	W	0	0	14	4	Sets width of HitMap pulses
20	HitMap_Width_Rd	R	0	0	14	4	Reads back value in the HitMap width register
21	HitMap_Deadtime_Wr	W	0	0	15	3	Sets stretch at end of HitMap pulses
22	HitMap_Deadtime_Rd	R	0	0	15	3	Sets stretch at end of HitMap pulses
23	Look_At_Me	W	0	1	4	16	GARC interface selection command (primary/secondary)
24	HitMap_Delay_Wr	W	0	1	8	5	Sets delay from Disc_In to HitMap pulse
25	HitMap_Delay_Rd	R	0	1	8	5	Reads back contents of the HitMap_Delay register
26	PHA_EN0_Wr	W	0	1	9	16	Enables and Disables PHA readout enable, channels 0 -15
27	PHA_EN0_Rd	R	0	1	9	16	Reads back contents of the PHA_EN0 register
28	VETO_EN0_Wr	W	0	1	10	16	Enables and Disables VETO signals, channels 0 -15
29	VETO_EN0_Rd	R	0	1	10	16	Reads back contents of the VETO_EN0 register
30	PHA_EN1_Wr	W	0	1	12	2	Enables and Disables PHA readout enable, channels 16 & 17
31	PHA_EN1_Rd	R	0	1	12	2	Reads back contents of the PHA_EN1 register
32	VETO_EN1_Wr	W	0	1	13	2	Enables and Disables VETO signals,

							channels 16 & 17
33	VETO_EN1_Rd	R	0	1	13	2	Reads back contents of the VETP_EN1 register
34	Max_PHA_Wr	W	0	1	15	5	Sets the Maximum number (limit) of PHA to be returned in a single event data packet
35	Max_PHA_Rd	R	0	1	15	5	Reads back the contents of the Max_PHA register
36	GARC_Mode_Wr	W	0	2	8	11	Sets values of GARC mode bits
37	GARC_Mode_Rd	R	0	2	8	11	Reads back the value of the GARC mode register
38	GARC_Status	R	0	2	9	6	Reads back the value of the GARC status register
39	GARC_Cmd_Reg	R	0	2	10	16	Reads back the value of the GARC command register
40	GARC_Diagnostic	R	0	2	11	16	Reads back the value of the GARC diagnostic register
41	GARC_Cmd_Rejects	R	0	2	12	8	Reads back the number of rejected commands since reset
42	FREE_Board_ID	R	0	2	13	8	Reads back the FREE board serial number
43	GARC_Version	R	0	2	14	3	Reads back the GARC ASIC version number
44	PHA_Thresh00_Wr	W	0	3	8	12	Writes PHA ZS threshold for channel 00
45	PHA_Thresh00_Rd	R	0	3	8	12	Reads back PHA ZS threshold register for channel 00
46	PHA_Thresh01_Wr	W	0	3	9	12	Writes PHA ZS threshold for channel 01
47	PHA_Thresh01_Rd	R	0	3	9	12	Reads back PHA ZS threshold register for channel 01
48	PHA_Thresh02_Wr	W	0	3	10	12	Writes PHA ZS threshold for channel 02
49	PHA_Thresh02_Rd	R	0	3	10	12	Reads back PHA ZS threshold register for channel 02
50	PHA_Thresh03_Wr	W	0	3	11	12	Writes PHA ZS threshold for channel 03
51	PHA_Thresh03_Rd	R	0	3	11	12	Reads back PHA ZS threshold register for channel 03
52	PHA_Thresh04_Wr	W	0	3	12	12	Writes PHA ZS threshold for channel 04
53	PHA_Thresh04_Rd	R	0	3	12	12	Reads back PHA ZS threshold register for channel 04
54	PHA_Thresh05_Wr	W	0	3	13	12	Writes PHA ZS threshold for channel 05
55	PHA_Thresh05_Rd	R	0	3	13	12	Reads back PHA ZS threshold register for channel 05
56	PHA_Thresh06_Wr	W	0	3	14	12	Writes PHA ZS threshold for channel 06
57	PHA_Thresh06_Rd	R	0	3	14	12	Reads back PHA ZS threshold register for channel 06
58	PHA_Thresh07_Wr	W	0	4	8	12	Writes PHA ZS threshold for channel 07
59	PHA_Thresh07_Rd	R	0	4	8	12	Reads back PHA ZS threshold register for channel 07
60	PHA_Thresh08_Wr	W	0	4	9	12	Writes PHA ZS threshold for channel 08
61	PHA_Thresh08_Rd	R	0	4	9	12	Reads back PHA ZS threshold register for channel 08
62	PHA_Thresh09_Wr	W	0	4	10	12	Writes PHA ZS threshold for channel 09
63	PHA_Thresh09_Rd	R	0	4	10	12	Reads back PHA ZS threshold register for channel 09
64	PHA_Thresh10_Wr	W	0	4	11	12	Writes PHA ZS threshold for channel 10
65	PHA_Thresh10_Rd	R	0	4	11	12	Reads back PHA ZS threshold register for channel 10
66	PHA_Thresh11_Wr	W	0	4	12	12	Writes PHA ZS threshold for channel 11
67	PHA_Thresh11_Rd	R	0	4	12	12	Reads back PHA ZS threshold register for channel 11
68	PHA_Thresh12_Wr	W	0	4	13	12	Writes PHA ZS threshold for channel 12
69	PHA_Thresh12_Rd	R	0	4	13	12	Reads back PHA ZS threshold register for channel 12
70	PHA_Thresh13_Wr	W	0	4	14	12	Writes PHA ZS threshold for channel 13

71	PHA_Thresh13_Rd	R	0	4	14	12	Reads back PHA ZS threshold register for channel 13
72	PHA_Thresh14_Wr	W	0	5	8	12	Writes PHA ZS threshold for channel 14
73	PHA_Thresh14_Rd	R	0	5	8	12	Reads back PHA ZS threshold register for channel 14
74	PHA_Thresh15_Wr	W	0	5	9	12	Writes PHA ZS threshold for channel 15
75	PHA_Thresh15_Rd	R	0	5	9	12	Reads back PHA ZS threshold register for channel 15
76	PHA_Thresh16_Wr	W	0	5	10	12	Writes PHA ZS threshold for channel 16
77	PHA_Thresh16_Rd	R	0	5	10	12	Reads back PHA ZS threshold register for channel 16
78	PHA_Thresh17_Wr	W	0	5	11	12	Writes PHA ZS threshold for channel 17
79	PHA_Thresh17_Rd	R	0	5	11	12	Reads back PHA ZS threshold register for channel 17
80	ADC_TACQ_Wr	W	0	5	12	6	Sets ADC Acquisition Time from Hold to Start of Conversion
81	ADC_TACQ_Rd	R	0	5	12	6	Reads back contents of the ADC_TACQ register
82	GAFE_Mode_Wr	W	1	GAFE Addr	0	16	Writes the GAFE mode register for the ASIC addressed
83	GAFE_Mode_Rd	R	1	GAFE Addr	0	16	Reads the GAFE mode register contents for the ASIC addressed
84	GAFE_DAC1_Wr	W	1	GAFE Addr	1	6	Writes the DAC1 register in the GAFE addressed
85	GAFE_DAC1_Rd	R	1	GAFE Addr	1	6	Reads back the contents of the DAC1 register in the addressed GAFE
86	GAFE_DAC2_Wr	W	1	GAFE Addr	2	6	Writes the DAC2 register in the GAFE addressed
87	GAFE_DAC2_Rd	R	1	GAFE Addr	2	6	Reads back the contents of the DAC2 register in the addressed GAFE
88	GAFE_DAC3_Wr	W	1	GAFE Addr	3	6	Writes the DAC3 register in the GAFE addressed
89	GAFE_DAC3_Rd	R	1	GAFE Addr	3	6	Reads back the contents of the DAC3 register in the addressed GAFE
90	GAFE_DAC4_Wr	W	1	GAFE Addr	4	6	Writes the DAC4 register in the GAFE addressed
91	GAFE_DAC4_Rd	R	1	GAFE Addr	4	6	Reads back the contents of the DAC4 register in the addressed GAFE
92	GAFE_DAC5_Wr	W	1	GAFE Addr	5	6	Writes the DAC5 register in the GAFE addressed
93	GAFE_DAC5_Rd	R	1	GAFE Addr	5	6	Reads back the contents of the DAC5 register in the addressed GAFE
94	GAFE_Version	R	1	GAFE Addr	6	3	Reads back the GAFE ASIC version
95	GAFE_Write_Ctr	R	1	GAFE Addr	7	8	Reads back the contents of the GAFE write counter register
96	GAFE_Reject_Ctr	R	1	GAFE Addr	8	8	Reads back the contents of the GAFE command reject register
97	GAFE_Cmd_Ctr	R	1	GAFE Addr	9	8	Reads back the contents of the GAFE command counter
98	GAFE_Chip_Addr	R	1	GAFE Addr	10	5	Reads back the hardwired address of a GAFE ASIC

#### 4.0 Testing the GARC Digital Logic

The GARC logic is based on a command-response protocol and requires an AEM or AEM simulator to access the logic functions. For each of the following commands, the proper GARC and/or GAFE response may be tested. Note that all GAFEs will respond to a broadcast write command (e.g., GAFE address of decimal 31), but a read command requires a unique GAFE address. The following test details the proper sequence for a single GARC ASIC. The steps are numbered for easier reference. The test setup required is detailed in the diagram below.



## GARC ASIC FUNCTIONAL TEST SETUP

### 4.1 Measurement of the GARC Bias Resistor Voltages

Verify that the GARC power supply is set to +3.3V. Using the GLAST GARC Test Board measure the following bias voltages on the resistors indicated.

At turn-on, measure the +3.3V current: \_\_\_\_\_  $\pm 10\text{mA}$

GARC Bias Signal	GARC Pin	Resistor Test Point	Expected Voltage	Measured Voltage
HLD_WOR_BIAS	104	R19	1.64	
BIAS_RCVR	156	R16	1.10	
BIAS_DRV_H	160	R15	1.53	
BIAS_DRV_L	169	R13	1.75	
LVDS_PRESET_ADJ	184	R20	1.52	

### 4.2 Initial Reset Test

This section will verify that GARC registers have been properly initialized during a reset command. The following test sequence of commands will perform this verification. This test will also verify that the GARC to AEM command link is functional.

1. Verify the external GARC Power On Reset pulse. Use the LabView GSE and send the Read All Values command.
2. Send the Veto\_Delay\_Rd command. The data field in the return data stream should be 5.
3. Send the HVBS\_Level\_Rd command. The data field in the return data stream should be 0.
4. Send the SAA\_Level\_Rd command. The data field in the return data stream should be 0.
5. Send the Hold\_Delay\_Rd command. The data field in the return data stream should be 28.
6. Send the Veto\_Width\_Rd command. The data field in the return data stream should be 2.
7. Send the HitMap\_Width\_Rd command. The data field in the return data stream should be 7.
8. Send the HitMap\_Deadtime\_Rd command. The data field in the return data stream should be 3.
9. Send the HitMap\_Delay\_Rd command. The data field in the return data stream should be 16.
10. Send the PHA\_En0\_Rd command. The data field in the return data stream should be 65535.
11. Send the PHA\_En1\_Rd command. The data field in the return data stream should be 3.
12. Send the Veto\_En0\_Rd command. The data field in the return data stream should be 65535.
13. Send the Veto\_En1\_Rd command. The data field in the return data stream should be 3.
14. Send the Max\_PHA\_Rd command. The data field in the return data stream should be 4.
15. Send the GARC\_Mode\_Rd command. The data field in the return data stream should be 768.
16. Send the GARC\_Status command. The data field in the return data stream should be 24.
17. Send the GARC\_Cmd\_Reg command. The data field in the return data stream should be 0.
18. Send the GARC\_Cmd\_Rejects command. The data field in the return data stream should be 0.
19. Send the GARC\_Version command. The data field in the return data stream should be 2 for GARC V2 and 3 for GARC V3.
20. Send the PHA\_Thresh00\_Rd command. The data field in the return data stream should be 1114.
21. Send the PHA\_Thresh01\_Rd command. The data field in the return data stream should be 1114.
22. Send the PHA\_Thresh02\_Rd command. The data field in the return data stream should be 1114.
23. Send the PHA\_Thresh03\_Rd command. The data field in the return data stream should be 1114.
24. Send the PHA\_Thresh04\_Rd command. The data field in the return data stream should be 1114.
25. Send the PHA\_Thresh05\_Rd command. The data field in the return data stream should be 1114.
26. Send the PHA\_Thresh06\_Rd command. The data field in the return data stream should be 1114.
27. Send the PHA\_Thresh07\_Rd command. The data field in the return data stream should be 1114.
28. Send the PHA\_Thresh08\_Rd command. The data field in the return data stream should be 1114.
29. Send the PHA\_Thresh09\_Rd command. The data field in the return data stream should be 1114.
30. Send the PHA\_Thresh10\_Rd command. The data field in the return data stream should be 1114.
31. Send the PHA\_Thresh11\_Rd command. The data field in the return data stream should be 1114.
32. Send the PHA\_Thresh12\_Rd command. The data field in the return data stream should be 1114.
33. Send the PHA\_Thresh13\_Rd command. The data field in the return data stream should be 1114.

34. Send the PHA\_Thresh14\_Rd command. The data field in the return data stream should be 1114.
35. Send the PHA\_Thresh15\_Rd command. The data field in the return data stream should be 1114.
36. Send the PHA\_Thresh16\_Rd command. The data field in the return data stream should be 1114.
37. Send the PHA\_Thresh17\_Rd command. The data field in the return data stream should be 1114.
38. Send the ADC\_TACQ\_Rd command. The data field in the return data stream should be 0.
39. Send the Trigger\_ZS command (this captures the FREE board ID). Verify the FREE board ID is 165 indicating the GARC Test board.
40. Send the GARC\_Reset command (Do Reset button). Verify that the FREE ID = 255.
41. Verify the status of the GARC registers as shown in steps 2 – 38 above.

If all steps above have verified correctly, the GARC reset function has been verified. A summary of the initial reset parameters for GARC and GAFE is detailed below.

Register	Initial Value (decimal)	Initial Value (hex)
Veto_Delay	5	5
HVBS_Level	0	0
SAA_Level	0	0
Hold_Delay	28	1C
Veto_Width	2	2
HitMap_Width	7	7
HitMap_Deptime	3	3
HitMap_Delay	16	10
PHA_EN0	65535	FFFF
PHA_EN1	3	3
VETO_EN0	65535	FFFF
VETO_EN1	3	3
Max_PHA	4	4
GARC_Mode	768	300
GARC_Status	24	18
Command_Register	0	0
GARC_Diagnostic	0	0
Cmd_Reject_Ctr	0	0
FREE_Board_ID	*	*
GARC_Version	1	1
PHA_Threshold_00	1114	45A
PHA_Threshold_01	1114	45A
PHA_Threshold_02	1114	45A
PHA_Threshold_03	1114	45A
PHA_Threshold_04	1114	45A
PHA_Threshold_05	1114	45A
PHA_Threshold_06	1114	45A
PHA_Threshold_07	1114	45A
PHA_Threshold_08	1114	45A
PHA_Threshold_09	1114	45A
PHA_Threshold_10	1114	45A
PHA_Threshold_11	1114	45A
PHA_Threshold_12	1114	45A
PHA_Threshold_13	1114	45A
PHA_Threshold_14	1114	45A
PHA_Threshold_15	1114	45A

PHA Threshold 16	1114	45A
PHA Threshold 17	1114	45A
ADC TACQ	0	0
GAFE Mode	48	30
GAFE DAC1	57	39
GAFE DAC2	38	26
GAFE DAC3	55	37
GAFE DAC4	32	20
GAFE DAC5	0	0
GAFE Wr Ctr	0	0
GAFE Reject Ctr	0	0

## 5.0 GARC Power Measurements

### 5.1 Measurement at the Nominal Power Supply Voltage

Verify that the GARC power supply is set to +3.30V. After initial power up, measure the +3.30V power supply current to the GARC in the following modes. Record these values in the table below.

1. Send the GARC\_Mode\_Wr command with a data argument of 768. Send the GARC\_Mode\_Rd command to verify. Both the primary and secondary LVDS VETO drivers should be enabled. Record the GARC current in the table below.
2. Send the GARC\_Mode\_Wr command with a data argument of 256. Send the GARC\_Mode\_Rd command to verify. Only the primary LVDS VETO drivers should now be enabled. Record the GARC current in the table below.
3. Send the GARC\_Mode\_Wr command with a data argument of 512. Send the GARC\_Mode\_Rd command to verify. Only the secondary LVDS VETO drivers should now be enabled. Record the GARC current in the table below.
4. Send the GARC\_Mode\_Wr command with a data argument of 0. Send the GARC\_Mode\_Rd command to verify. None of the LVDS VETO drivers should now be enabled. Record the GARC current in the table below.

GARC Mode	GARC_Mode_Wr Data Argument	+3.3V Current Measured (mA)	+3.3V Current Expected (mA)
LVDS "A" Drivers Enabled LVDS "B" Drivers Enabled	768		185 ± 10mA
LVDS "A" Drivers Enabled LVDS "B" Drivers Disabled	256		135 ± 10mA
LVDS "A" Drivers Disabled LVDS "B" Drivers Enabled	512		135 ± 10mA
LVDS "A" Drivers Disabled LVDS "B" Drivers Disabled	0		80 ± 10mA

### 5.2 Measurement at the Minimum Power Supply Voltage

Verify that the GARC power supply is set to +3.0V. After initial power up, measure the +3.0V power supply current to the GARC in the following modes. Record these values in the table below.

1. Send the GARC\_Mode\_Wr command with a data argument of 768. Send the GARC\_Mode\_Rd command to verify. Both the primary and secondary LVDS VETO drivers should be enabled. Record the GARC current in the table below.
2. Send the GARC\_Mode\_Wr command with a data argument of 256. Send the GARC\_Mode\_Rd command to verify. Only the primary LVDS VETO drivers should now be enabled. Record the GARC current in the table below.
3. Send the GARC\_Mode\_Wr command with a data argument of 512. Send the GARC\_Mode\_Rd command to verify. Only the secondary LVDS VETO drivers should now be enabled. Record the GARC current in the table below.
4. Send the GARC\_Mode\_Wr command with a data argument of 0. Send the GARC\_Mode\_Rd command to verify. None of the LVDS VETO drivers should now be enabled. Record the GARC current in the table below.

GARC Mode	GARC_Mode_Wr Data Argument	+3.0V Current Measured (mA)	+3.0V Current Expected (mA)
LVDS "A" Drivers Enabled LVDS "B" Drivers Enabled	768		160 ± 10mA
LVDS "A" Drivers Enabled LVDS "B" Drivers Disabled	256		115 ± 10mA
LVDS "A" Drivers Disabled LVDS "B" Drivers Enabled	512		115 ± 10mA
LVDS "A" Drivers Disabled LVDS "B" Drivers Disabled	0		65 ± 10mA

### 5.3 Measurement at the Maximum Power Supply Voltage

Verify that the GARC power supply is set to +3.6V. After initial power up, measure the +3.6V power supply current to the GARC in the following modes. Record these values in the table below.

1. Send the GARC\_Mode\_Wr command with a data argument of 768. Send the GARC\_Mode\_Rd command to verify. Both the primary and secondary LVDS VETO drivers should be enabled. Record the GARC current in the table below.
2. Send the GARC\_Mode\_Wr command with a data argument of 256. Send the GARC\_Mode\_Rd command to verify. Only the primary LVDS VETO drivers should now be enabled. Record the GARC current in the table below.
3. Send the GARC\_Mode\_Wr command with a data argument of 512. Send the GARC\_Mode\_Rd command to verify. Only the secondary LVDS VETO drivers should now be enabled. Record the GARC current in the table below.
4. Send the GARC\_Mode\_Wr command with a data argument of 0. Send the GARC\_Mode\_Rd command to verify. None of the LVDS VETO drivers should now be enabled. Record the GARC current in the table below.

GARC Mode	GARC_Mode_Wr Data Argument	+3.6V Current Measured (mA)	+3.6V Current Expected (mA)
LVDS "A" Drivers Enabled LVDS "B" Drivers Enabled	768		215 ± 10mA
LVDS "A" Drivers Enabled LVDS "B" Drivers Disabled	256		155 ± 10mA
LVDS "A" Drivers Disabled LVDS "B" Drivers Enabled	512		155 ± 10mA
LVDS "A" Drivers Disabled LVDS "B" Drivers Disabled	0		95 ± 10mA

5. Reset the GARC power supply to +3.3V.
6. Send the GARC\_Reset command to return the GARC to the initial power-on configuration.

## 6.0 GARC Register Read/Write and PHA Readout Tests

This section tests the proper functioning of each bit of the commandable registers in the GARC. The intent is to toggle each bit in a variety of patterns to ensure that all bits are addressable and that there is no stuck-at-fault condition. This can be accomplished using the LabView GSE software by doing the Register Test 3 times with all GAFE registers enabled. If done via the automated procedure, the next section will be 6.40. The following GARC registers will be tested in this section:

<b>GARC Register Name</b>	<b>Register Width (bits)</b>	<b>Register Type</b>
Veto_Delay	5	Read/Write
HVBS_Level	12	Read/Write
SAA_Level	12	Read/Write
Hold_Delay	7	Read/Write
Veto_Width	3	Read/Write
HitMap_Width	4	Read/Write
HitMap_Deadtime	3	Read/Write
HitMap_Delay	5	Read/Write
PHA_En0	16	Read/Write
PHA_En1	2	Read/Write
VETO_En0	16	Read/Write
VETO_En1	2	Read/Write
Max_PHA	5	Read/Write
GARC_Mode	11	Read/Write
GARC_Status	6	Read Only
Command_Register	16	Read Only
GARC_Diagnostic	16	Read Only
Cmd_Reject_Counter	8	Read Only
FREE_Board_ID	8	Read Only
GARC_Version	3	Read Only
PHA_Threshold_00	16	Read/Write
PHA_Threshold_01	16	Read/Write
PHA_Threshold_02	16	Read/Write
PHA_Threshold_03	16	Read/Write
PHA_Threshold_04	16	Read/Write
PHA_Threshold_05	16	Read/Write
PHA_Threshold_06	16	Read/Write
PHA_Threshold_07	16	Read/Write
PHA_Threshold_08	16	Read/Write
PHA_Threshold_09	16	Read/Write
PHA_Threshold_10	16	Read/Write
PHA_Threshold_11	16	Read/Write
PHA_Threshold_12	16	Read/Write
PHA_Threshold_13	16	Read/Write
PHA_Threshold_14	16	Read/Write
PHA_Threshold_15	16	Read/Write
PHA_Threshold_16	16	Read/Write
PHA_Threshold_17	16	Read/Write
ADC_TACQ	6	Read/Write

**6.1 Veto Delay Register Test**

1. Send the Veto\_Delay\_Wr command with a data argument of 5'h0 (0). Send the Veto\_Delay\_Rd command and read back this commanded data argument.
2. Send the Veto\_Delay\_Wr command with a data argument of 5'h15 (21). Send the Veto\_Delay\_Rd command and read back this commanded data argument.
3. Send the Veto\_Delay\_Wr command with a data argument of 5'h0A (10). Send the Veto\_Delay\_Rd command and read back this commanded data argument.
4. Send the Veto\_Delay\_Wr command with a data argument of 5'h1F (31). Send the Veto\_Delay\_Rd command and read back this commanded data argument.
5. Send the Veto\_Delay\_Wr command with a data argument of 5'h5 (5). Send the Veto\_Delay\_Rd command and read back this commanded data argument.

**6.2 HVBS Level Register Test**

1. Send the HVBS\_Level\_Wr command with a data argument of 12'h0 (0). Send the HVBS\_Level\_Rd command and read back this commanded data argument.
2. Send the HVBS\_Level\_Wr command with a data argument of 12'h555 (1365). Send the HVBS\_Level\_Rd command and read back this commanded data argument.
3. Send the HVBS\_Level\_Wr command with a data argument of 12'hAAA (2730). Send the HVBS\_Level\_Rd command and read back this commanded data argument.
4. Send the HVBS\_Level\_Wr command with a data argument of 12'hFFF (4095). Send the HVBS\_Level\_Rd command and read back this commanded data argument.
5. Send the HVBS\_Level\_Wr command with a data argument of 12'h0 (0). Send the HVBS\_Level\_Rd command and read back this commanded data argument.

**6.3 SAA Level Register Test**

1. Send the SAA\_Level\_Wr command with a data argument of 12'h0 (0). Send the SAA\_Level\_Rd command and read back this commanded data argument.
2. Send the SAA\_Level\_Wr command with a data argument of 12'h555 (1365). Send the SAA\_Level\_Rd command and read back this commanded data argument.
3. Send the SAA\_Level\_Wr command with a data argument of 12'hAAA (2730). Send the SAA\_Level\_Rd command and read back this commanded data argument.
4. Send the SAA\_Level\_Wr command with a data argument of 12'hFFF (4095). Send the SAA\_Level\_Rd command and read back this commanded data argument.
5. Send the SAA\_Level\_Wr command with a data argument of 12'h0 (0). Send the SAA\_Level\_Rd command and read back this commanded data argument.

**6.4 Hold Delay Register Test**

1. Send the Hold\_Delay\_Wr command with a data argument of 7'h0 (0). Send the Hold\_Delay\_Rd command and read back this commanded data argument.

2. Send the Hold\_Delay\_Wr command with a data argument of 7'h55 (85). Send the Hold\_Delay\_Rd command and read back this commanded data argument.
3. Send the Hold\_Delay\_Wr command with a data argument of 7'h2A (42). Send the Hold\_Delay\_Rd command and read back this commanded data argument.
4. Send the Hold\_Delay\_Wr command with a data argument of 7'h7F (127). Send the Hold\_Delay\_Rd command and read back this commanded data argument.
5. Send the Hold\_Delay\_Wr command with a data argument of 7'h1C (28). Send the Hold\_Delay\_Rd command and read back this commanded data argument.

#### **6.5 Veto Width Register Test**

1. Send the Veto\_Width\_Wr command with a data argument of 3'h0 (0). Send the Veto\_Width\_Rd command and read back this commanded data argument.
2. Send the Veto\_Width\_Wr command with a data argument of 3'h5 (5). Send the Veto\_Width\_Rd command and read back this commanded data argument.
3. Send the Veto\_Width\_Wr command with a data argument of 3'h7 (7). Send the Veto\_Width\_Rd command and read back this commanded data argument.
4. Send the Veto\_Width\_Wr command with a data argument of 3'h2 (2). Send the Veto\_Width\_Rd command and read back this commanded data argument.

#### **6.6 HitMap Width Register Test**

1. Send the HitMap\_Width\_Wr command with a data argument of 4'h0 (0). Send the HitMap\_Width\_Rd command and read back this commanded data argument.
2. Send the HitMap\_Width\_Wr command with a data argument of 4'h5 (5). Send the HitMap\_Width\_Rd command and read back this commanded data argument.
3. Send the HitMap\_Width\_Wr command with a data argument of 4'hA (10). Send the HitMap\_Width\_Rd command and read back this commanded data argument.
4. Send the HitMap\_Width\_Wr command with a data argument of 4'hF (15). Send the HitMap\_Width\_Rd command and read back this commanded data argument.
5. Send the HitMap\_Width\_Wr command with a data argument of 4'h7 (7). Send the HitMap\_Width\_Rd command and read back this commanded data argument.

#### **6.7 HitMap Deadtime Register Test**

1. Send the HitMap\_Deadtime\_Wr command with a data argument of 3'h0 (0). Send the HitMap\_Deadtime\_Rd command and read back this commanded data argument.
2. Send the HitMap\_Deadtime\_Wr command with a data argument of 3'h5 (5). Send the HitMap\_Deadtime\_Rd command and read back this commanded data argument.
3. Send the HitMap\_Deadtime\_Wr command with a data argument of 3'h2 (2). Send the HitMap\_Deadtime\_Rd command and read back this commanded data argument.
4. Send the HitMap\_Deadtime\_Wr command with a data argument of 3'h7 (7). Send the HitMap\_Deadtime\_Rd command and read back this commanded data argument.
5. Send the HitMap\_Deadtime\_Wr command with a data argument of 3'h3 (3). Send the HitMap\_Deadtime\_Rd command and read back this commanded data argument.

**6.8 HitMap Delay Register Test**

1. Send the HitMap\_Delay\_Wr command with a data argument of 5'h0 (0). Send the HitMap\_Delay\_Rd command and read back this commanded data argument.
2. Send the HitMap\_Delay\_Wr command with a data argument of 5'h15 (21). Send the HitMap\_Delay\_Rd command and read back this commanded data argument.
3. Send the HitMap\_Delay\_Wr command with a data argument of 5'h0A (10). Send the HitMap\_Delay\_Rd command and read back this commanded data argument.
4. Send the HitMap\_Delay\_Wr command with a data argument of 5'h1F (31). Send the HitMap\_Delay\_Rd command and read back this commanded data argument.
5. Send the HitMap\_Delay\_Wr command with a data argument of 5'h10 (16). Send the HitMap\_Delay\_Rd command and read back this commanded data argument.

**6.9 PHA En0 Register Test**

1. Send the PHA\_En0\_Wr command with a data argument of 16'h0 (0). Send the PHA\_En0\_Rd command and read back this commanded data argument.
2. Send the PHA\_En0\_Wr command with a data argument of 16'h5555 (21845). Send the PHA\_En0\_Rd command and read back this commanded data argument.
3. Send the PHA\_En0\_Wr command with a data argument of 16'hAAAA (43690). Send the PHA\_En0\_Rd command and read back this commanded data argument.
4. Send the PHA\_En0\_Wr command with a data argument of 16'hFFFF (65535). Send the PHA\_En0\_Rd command and read back this commanded data argument.

**6.10 Veto En0 Register Test**

1. Send the VETO\_En0\_Wr command with a data argument of 16'h0 (0). Send the VETO\_En0\_Rd command and read back this commanded data argument.
2. Send the VETO\_En0\_Wr command with a data argument of 16'h5555 (21845). Send the VETO\_En0\_Rd command and read back this commanded data argument.
3. Send the VETO\_En0\_Wr command with a data argument of 16'hAAAA (43690). Send the VETO\_En0\_Rd command and read back this commanded data argument.
4. Send the VETO\_En0\_Wr command with a data argument of 16'hFFFF (65535). Send the VETO\_En0\_Rd command and read back this commanded data argument.

**6.11 PHA En1 Register Test**

1. Send the PHA\_En1\_Wr command with a data argument of 2'h0 (0). Send the PHA\_En1\_Rd command and read back this commanded data argument.
2. Send the PHA\_En1\_Wr command with a data argument of 2'h1 (1). Send the PHA\_En1\_Rd command and read back this commanded data argument.
3. Send the PHA\_En1\_Wr command with a data argument of 2'h2 (2). Send the PHA\_En1\_Rd command and read back this commanded data argument.

4. Send the PHA\_En1\_Wr command with a data argument of 2'h3 (3). Send the PHA\_En1\_Rd command and read back this commanded data argument.

#### **6.12 Veto En1 Register Test**

1. Send the VETO\_En1\_Wr command with a data argument of 2'h0 (0). Send the VETO\_En1\_Rd command and read back this commanded data argument.
2. Send the VETO\_En1\_Wr command with a data argument of 2'h1 (1). Send the VETO\_En1\_Rd command and read back this commanded data argument.
3. Send the VETO\_En1\_Wr command with a data argument of 2'h2 (2). Send the VETO\_En1\_Rd command and read back this commanded data argument.
4. Send the VETO\_En1\_Wr command with a data argument of 2'h3 (3). Send the VETO\_En1\_Rd command and read back this commanded data argument.

#### **6.13 MaxPHA Register Test**

1. Send the MaxPHA\_Wr command with a data argument of 5'h0 (0). Send the MaxPHA\_Rd command and read back this commanded data argument.
2. Send the MaxPHA\_Wr command with a data argument of 5'h15 (21). Send the MaxPHA\_Rd command and read back this commanded data argument.
3. Send the MaxPHA\_Wr command with a data argument of 5'h0A (10). Send the MaxPHA\_Rd command and read back this commanded data argument.
4. Send the MaxPHA\_Wr command with a data argument of 5'h1F (31). Send the MaxPHA\_Rd command and read back this commanded data argument.
5. Send the MaxPHA\_Wr command with a data argument of 5'h4 (4). Send the MaxPHA\_Rd command and read back this commanded data argument.

#### **6.14 GARC Mode Register Test**

**\*\* Note this sequence must be followed exactly to preclude placing the GARC into the undesired mode of sending return data with incorrect parity.**

1. Send the GARC\_Mode\_Wr command with a data argument of 11'h2 (2). Send the GARC\_Mode\_Rd command and read back this commanded data argument.
2. Send the GARC\_Mode\_Wr command with a data argument of 11'h4 (4). Send the GARC\_Mode\_Rd command and read back this commanded data argument.
3. Send the GARC\_Mode\_Wr command with a data argument of 11'h8 (8). Send the GARC\_Mode\_Rd command and read back this commanded data argument.
4. Send the GARC\_Mode\_Wr command with a data argument of 11'h10 (16). Send the GARC\_Mode\_Rd command and read back this commanded data argument.
5. Send the GARC\_Mode\_Wr command with a data argument of 11'h20 (32). Send the GARC\_Mode\_Rd command and read back this commanded data argument.
6. Send the GARC\_Mode\_Wr command with a data argument of 11'h40 (64). Send the GARC\_Mode\_Rd command and read back this commanded data argument.

7. Send the GARC\_Mode\_Wr command with a data argument of 11'h80 (128). Send the GARC\_Mode\_Rd command and read back this commanded data argument.
8. Send the GARC\_Mode\_Wr command with a data argument of 11'h100 (256). Send the GARC\_Mode\_Rd command and read back this commanded data argument.
9. Send the GARC\_Mode\_Wr command with a data argument of 11'h200 (512). Send the GARC\_Mode\_Rd command and read back this commanded data argument.
10. Send the GARC\_Mode\_Wr command with a data argument of 11'h400 (1024). Send the GARC\_Mode\_Rd command and read back this commanded data argument.
11. Send the GARC\_Mode\_Wr command with a data argument of 11'h300 (768). Send the GARC\_Mode\_Rd command and read back this commanded data argument.

#### **6.15 PHA Threshold Channel 00 Register Test**

1. Send the PHA\_Thresh00\_Wr command with a data argument of 12'h0 (0). Send the PHA\_Thresh00\_Rd command and read back this commanded data argument.
2. Send the PHA\_Thresh00\_Wr command with a data argument of 12'h555 (21845). Send the PHA\_Thresh00\_Rd command and read back this commanded data argument.
3. Send the PHA\_Thresh00\_Wr command with a data argument of 16'hAAA (43690). Send the PHA\_Thresh00\_Rd command and read back this commanded data argument.
4. Send the PHA\_Thresh00\_Wr command with a data argument of 16'hFFF (65536). Send the PHA\_Thresh00\_Rd command and read back this commanded data argument.
5. Send the PHA\_Thresh00\_Wr command with a data argument of 16'h45A (1114). Send the PHA\_Thresh00\_Rd command and read back this commanded data argument.

#### **6.16 PHA Threshold Channel 01 Register Test**

1. Send the PHA\_Thresh01\_Wr command with a data argument of 16'h0 (0). Send the PHA\_Thresh01\_Rd command and read back this commanded data argument.
2. Send the PHA\_Thresh01\_Wr command with a data argument of 16'h5555 (21845). Send the PHA\_Thresh01\_Rd command and read back this commanded data argument.
3. Send the PHA\_Thresh01\_Wr command with a data argument of 16'hAAAA (43690). Send the PHA\_Thresh01\_Rd command and read back this commanded data argument.
4. Send the PHA\_Thresh01\_Wr command with a data argument of 16'hAAAA (65536). Send the PHA\_Thresh01\_Rd command and read back this commanded data argument.
5. Send the PHA\_Thresh01\_Wr command with a data argument of 16'h45A (1114). Send the PHA\_Thresh01\_Rd command and read back this commanded data argument.

#### **6.17 PHA Threshold Channel 02 Register Test**

1. Send the PHA\_Thresh02\_Wr command with a data argument of 16'h0 (0). Send the PHA\_Thresh02\_Rd command and read back this commanded data argument.
2. Send the PHA\_Thresh02\_Wr command with a data argument of 16'h5555 (21845). Send the PHA\_Thresh02\_Rd command and read back this commanded data argument.
3. Send the PHA\_Thresh02\_Wr command with a data argument of 16'hAAAA (43690). Send the PHA\_Thresh02\_Rd command and read back this commanded data argument.

4. Send the PHA\_Thresh02\_Wr command with a data argument of 16'hAAAA (65536). Send the PHA\_Thresh02\_Rd command and read back this commanded data argument.
5. Send the PHA\_Thresh02\_Wr command with a data argument of 16'h45A (1114). Send the PHA\_Thresh02\_Rd command and read back this commanded data argument.

#### **6.18 PHA Threshold Channel 03 Register Test**

1. Send the PHA\_Thresh03\_Wr command with a data argument of 16'h0 (0). Send the PHA\_Thresh03\_Rd command and read back this commanded data argument.
2. Send the PHA\_Thresh03\_Wr command with a data argument of 16'h5555 (21845). Send the PHA\_Thresh03\_Rd command and read back this commanded data argument.
3. Send the PHA\_Thresh03\_Wr command with a data argument of 16'hAAAA (43690). Send the PHA\_Thresh03\_Rd command and read back this commanded data argument.
4. Send the PHA\_Thresh03\_Wr command with a data argument of 16'hAAAA (65536). Send the PHA\_Thresh03\_Rd command and read back this commanded data argument.
5. Send the PHA\_Thresh03\_Wr command with a data argument of 16'h45A (1114). Send the PHA\_Thresh03\_Rd command and read back this commanded data argument.

#### **6.19 PHA Threshold Channel 04 Register Test**

1. Send the PHA\_Thresh04\_Wr command with a data argument of 16'h0 (0). Send the PHA\_Thresh04\_Rd command and read back this commanded data argument.
2. Send the PHA\_Thresh04\_Wr command with a data argument of 16'h5555 (21845). Send the PHA\_Thresh04\_Rd command and read back this commanded data argument.
3. Send the PHA\_Thresh04\_Wr command with a data argument of 16'hAAAA (43690). Send the PHA\_Thresh04\_Rd command and read back this commanded data argument.
4. Send the PHA\_Thresh04\_Wr command with a data argument of 16'hAAAA (65536). Send the PHA\_Thresh04\_Rd command and read back this commanded data argument.
5. Send the PHA\_Thresh04\_Wr command with a data argument of 16'h45A (1114). Send the PHA\_Thresh04\_Rd command and read back this commanded data argument.

#### **6.20 PHA Threshold Channel 05 Register Test**

1. Send the PHA\_Thresh05\_Wr command with a data argument of 16'h0 (0). Send the PHA\_Thresh05\_Rd command and read back this commanded data argument.
2. Send the PHA\_Thresh05\_Wr command with a data argument of 16'h5555 (21845). Send the PHA\_Thresh05\_Rd command and read back this commanded data argument.
3. Send the PHA\_Thresh05\_Wr command with a data argument of 16'hAAAA (43690). Send the PHA\_Thresh05\_Rd command and read back this commanded data argument.
4. Send the PHA\_Thresh05\_Wr command with a data argument of 16'hAAAA (65536). Send the PHA\_Thresh05\_Rd command and read back this commanded data argument.
5. Send the PHA\_Thresh05\_Wr command with a data argument of 16'h45A (1114). Send the PHA\_Thresh05\_Rd command and read back this commanded data argument.

**6.21 PHA Threshold Channel 06 Register Test**

1. Send the PHA\_Thresh06\_Wr command with a data argument of 16'h0 (0). Send the PHA\_Thresh06\_Rd command and read back this commanded data argument.
2. Send the PHA\_Thresh06\_Wr command with a data argument of 16'h5555 (21845). Send the PHA\_Thresh06\_Rd command and read back this commanded data argument.
3. Send the PHA\_Thresh06\_Wr command with a data argument of 16'hAAAA (43690). Send the PHA\_Thresh06\_Rd command and read back this commanded data argument.
4. Send the PHA\_Thresh06\_Wr command with a data argument of 16'hAAAA (65536). Send the PHA\_Thresh06\_Rd command and read back this commanded data argument.
5. Send the PHA\_Thresh06\_Wr command with a data argument of 16'h45A (1114). Send the PHA\_Thresh06\_Rd command and read back this commanded data argument.

**6.22 PHA Threshold Channel 07 Register Test**

1. Send the PHA\_Thresh07\_Wr command with a data argument of 16'h0 (0). Send the PHA\_Thresh07\_Rd command and read back this commanded data argument.
2. Send the PHA\_Thresh07\_Wr command with a data argument of 16'h5555 (21845). Send the PHA\_Thresh07\_Rd command and read back this commanded data argument.
3. Send the PHA\_Thresh07\_Wr command with a data argument of 16'hAAAA (43690). Send the PHA\_Thresh07\_Rd command and read back this commanded data argument.
4. Send the PHA\_Thresh07\_Wr command with a data argument of 16'hAAAA (65536). Send the PHA\_Thresh07\_Rd command and read back this commanded data argument.
5. Send the PHA\_Thresh07\_Wr command with a data argument of 16'h45A (1114). Send the PHA\_Thresh07\_Rd command and read back this commanded data argument.

**6.23 PHA Threshold Channel 08 Register Test**

1. Send the PHA\_Thresh08\_Wr command with a data argument of 16'h0 (0). Send the PHA\_Thresh08\_Rd command and read back this commanded data argument.
2. Send the PHA\_Thresh08\_Wr command with a data argument of 16'h5555 (21845). Send the PHA\_Thresh08\_Rd command and read back this commanded data argument.
3. Send the PHA\_Thresh08\_Wr command with a data argument of 16'hAAAA (43690). Send the PHA\_Thresh08\_Rd command and read back this commanded data argument.
4. Send the PHA\_Thresh08\_Wr command with a data argument of 16'hAAAA (65536). Send the PHA\_Thresh08\_Rd command and read back this commanded data argument.
5. Send the PHA\_Thresh08\_Wr command with a data argument of 16'h45A (1114). Send the PHA\_Thresh08\_Rd command and read back this commanded data argument.

**6.24 PHA Threshold Channel 09 Register Test**

1. Send the PHA\_Thresh09\_Wr command with a data argument of 16'h0 (0). Send the PHA\_Thresh09\_Rd command and read back this commanded data argument.
2. Send the PHA\_Thresh09\_Wr command with a data argument of 16'h5555 (21845). Send the PHA\_Thresh09\_Rd command and read back this commanded data argument.

3. Send the PHA\_Thresh09\_Wr command with a data argument of 16'hAAAA (43690). Send the PHA\_Thresh09\_Rd command and read back this commanded data argument.
4. Send the PHA\_Thresh09\_Wr command with a data argument of 16'hAAAA (65536). Send the PHA\_Thresh09\_Rd command and read back this commanded data argument.
5. Send the PHA\_Thresh09\_Wr command with a data argument of 16'h45A (1114). Send the PHA\_Thresh09\_Rd command and read back this commanded data argument.

#### **6.25 PHA Threshold Channel 10 Register Test**

1. Send the PHA\_Thresh10\_Wr command with a data argument of 16'h0 (0). Send the PHA\_Thresh10\_Rd command and read back this commanded data argument.
2. Send the PHA\_Thresh10\_Wr command with a data argument of 16'h5555 (21845). Send the PHA\_Thresh10\_Rd command and read back this commanded data argument.
3. Send the PHA\_Thresh10\_Wr command with a data argument of 16'hAAAA (43690). Send the PHA\_Thresh10\_Rd command and read back this commanded data argument.
4. Send the PHA\_Thresh10\_Wr command with a data argument of 16'hAAAA (65536). Send the PHA\_Thresh10\_Rd command and read back this commanded data argument.
5. Send the PHA\_Thresh10\_Wr command with a data argument of 16'h45A (1114). Send the PHA\_Thresh10\_Rd command and read back this commanded data argument.

#### **6.26 PHA Threshold Channel 11 Register Test**

1. Send the PHA\_Thresh11\_Wr command with a data argument of 16'h0 (0). Send the PHA\_Thresh11\_Rd command and read back this commanded data argument.
2. Send the PHA\_Thresh11\_Wr command with a data argument of 16'h5555 (21845). Send the PHA\_Thresh11\_Rd command and read back this commanded data argument.
3. Send the PHA\_Thresh11\_Wr command with a data argument of 16'hAAAA (43690). Send the PHA\_Thresh11\_Rd command and read back this commanded data argument.
4. Send the PHA\_Thresh11\_Wr command with a data argument of 16'hAAAA (65536). Send the PHA\_Thresh11\_Rd command and read back this commanded data argument.
5. Send the PHA\_Thresh11\_Wr command with a data argument of 16'h45A (1114). Send the PHA\_Thresh11\_Rd command and read back this commanded data argument.

#### **6.27 PHA Threshold Channel 12 Register Test**

1. Send the PHA\_Thresh12\_Wr command with a data argument of 16'h0 (0). Send the PHA\_Thresh12\_Rd command and read back this commanded data argument.
2. Send the PHA\_Thresh12\_Wr command with a data argument of 16'h5555 (21845). Send the PHA\_Thresh12\_Rd command and read back this commanded data argument.
3. Send the PHA\_Thresh12\_Wr command with a data argument of 16'hAAAA (43690). Send the PHA\_Thresh12\_Rd command and read back this commanded data argument.
4. Send the PHA\_Thresh12\_Wr command with a data argument of 16'hAAAA (65536). Send the PHA\_Thresh12\_Rd command and read back this commanded data argument.
5. Send the PHA\_Thresh12\_Wr command with a data argument of 16'h45A (1114). Send the PHA\_Thresh12\_Rd command and read back this commanded data argument.

**6.28 PHA Threshold Channel 13 Register Test**

1. Send the PHA\_Thresh13\_Wr command with a data argument of 16'h0 (0). Send the PHA\_Thresh13\_Rd command and read back this commanded data argument.
2. Send the PHA\_Thresh13\_Wr command with a data argument of 16'h5555 (21845). Send the PHA\_Thresh13\_Rd command and read back this commanded data argument.
3. Send the PHA\_Thresh13\_Wr command with a data argument of 16'hAAAA (43690). Send the PHA\_Thresh13\_Rd command and read back this commanded data argument.
4. Send the PHA\_Thresh13\_Wr command with a data argument of 16'hAAAA (65536). Send the PHA\_Thresh13\_Rd command and read back this commanded data argument.
5. Send the PHA\_Thresh13\_Wr command with a data argument of 16'h45A (1114). Send the PHA\_Thresh13\_Rd command and read back this commanded data argument.

**6.29 PHA Threshold Channel 14 Register Test**

1. Send the PHA\_Thresh14\_Wr command with a data argument of 16'h0 (0). Send the PHA\_Thresh14\_Rd command and read back this commanded data argument.
2. Send the PHA\_Thresh14\_Wr command with a data argument of 16'h5555 (21845). Send the PHA\_Thresh14\_Rd command and read back this commanded data argument.
3. Send the PHA\_Thresh14\_Wr command with a data argument of 16'hAAAA (43690). Send the PHA\_Thresh14\_Rd command and read back this commanded data argument.
4. Send the PHA\_Thresh14\_Wr command with a data argument of 16'hAAAA (65536). Send the PHA\_Thresh14\_Rd command and read back this commanded data argument.
5. Send the PHA\_Thresh14\_Wr command with a data argument of 16'h45A (1114). Send the PHA\_Thresh14\_Rd command and read back this commanded data argument.

**6.30 PHA Threshold Channel 15 Register Test**

1. Send the PHA\_Thresh15\_Wr command with a data argument of 16'h0 (0). Send the PHA\_Thresh15\_Rd command and read back this commanded data argument.
2. Send the PHA\_Thresh15\_Wr command with a data argument of 16'h5555 (21845). Send the PHA\_Thresh15\_Rd command and read back this commanded data argument.
3. Send the PHA\_Thresh15\_Wr command with a data argument of 16'hAAAA (43690). Send the PHA\_Thresh15\_Rd command and read back this commanded data argument.
4. Send the PHA\_Thresh15\_Wr command with a data argument of 16'hAAAA (65536). Send the PHA\_Thresh15\_Rd command and read back this commanded data argument.
5. Send the PHA\_Thresh15\_Wr command with a data argument of 16'h45A (1114). Send the PHA\_Thresh15\_Rd command and read back this commanded data argument.

**6.31 PHA Threshold Channel 16 Register Test**

1. Send the PHA\_Thresh16\_Wr command with a data argument of 16'h0 (0). Send the PHA\_Thresh16\_Rd command and read back this commanded data argument.

2. Send the PHA\_Thresh16\_Wr command with a data argument of 16'h5555 (21845). Send the PHA\_Thresh16\_Rd command and read back this commanded data argument.
3. Send the PHA\_Thresh16\_Wr command with a data argument of 16'hAAAA (43690). Send the PHA\_Thresh16\_Rd command and read back this commanded data argument.
4. Send the PHA\_Thresh16\_Wr command with a data argument of 16'hAAAA (65536). Send the PHA\_Thresh16\_Rd command and read back this commanded data argument.
5. Send the PHA\_Thresh16\_Wr command with a data argument of 16'h45A (1114). Send the PHA\_Thresh16\_Rd command and read back this commanded data argument.

### **6.32 PHA Threshold Channel 17 Register Test**

1. Send the PHA\_Thresh17\_Wr command with a data argument of 16'h0 (0). Send the PHA\_Thresh17\_Rd command and read back this commanded data argument.
2. Send the PHA\_Thresh17\_Wr command with a data argument of 16'h5555 (21845). Send the PHA\_Thresh17\_Rd command and read back this commanded data argument.
3. Send the PHA\_Thresh17\_Wr command with a data argument of 16'hAAAA (43690). Send the PHA\_Thresh17\_Rd command and read back this commanded data argument.
4. Send the PHA\_Thresh17\_Wr command with a data argument of 16'hAAAA (65536). Send the PHA\_Thresh17\_Rd command and read back this commanded data argument.
5. Send the PHA\_Thresh17\_Wr command with a data argument of 16'h45A (1114). Send the PHA\_Thresh17\_Rd command and read back this commanded data argument.

### **6.33 ADC TACQ Register Test**

1. Send the ADC\_TACQ\_Wr command with a data argument of 6'h0 (0). Send the ADC\_TACQ\_Rd command and read back this commanded data argument.
2. Send the ADC\_TACQ\_Wr command with a data argument of 6'h15 (21). Send the ADC\_TACQ\_Rd command and read back this commanded data argument.
3. Send the ADC\_TACQ\_Wr command with a data argument of 6'h2A (42). Send the ADC\_TACQ\_Rd command and read back this commanded data argument.
4. Send the ADC\_TACQ\_Wr command with a data argument of 6'h3F (63). Send the ADC\_TACQ\_Rd command and read back this commanded data argument.
5. Send the ADC\_TACQ\_Wr command with a data argument of 6'h0 (0). Send the ADC\_TACQ\_Rd command and read back this commanded data argument.
6. Send the GARC\_Reset command to ensure all registers are at the proper initial value.

### **6.34 GAFE ASICs Mode Register Test**

This section tests the proper functioning of the GAFE mode register. Repeat for all GAFE ASICs present using valid GAFE addressing.

1. Send the GAFE\_Mode\_Write command to the GAFE being tested with a data field of 16'h00 (0).
2. Send the GAFE\_Mode\_Read command. The data field in the return data stream should be 16'h00 (0).
3. Send the GAFE\_Mode\_Write command to the GAFE being tested with a data field of 16'hFF (255).

4. Send the GAFE\_Mode\_Read command. The data field in the return data stream should be 16'hFF (255).
5. Send the GAFE\_Mode\_Write command to the GAFE being tested with a data field of 16'h30 (48).
6. Send the GAFE\_Mode\_Read command. The data field in the return data stream should be 16'h30 (48).

### **6.35 GAFE ASIC VETO DAC Register Test**

This section tests the proper functioning of the GAFE DAC #1 register. Repeat for all GAFE ASICs present, using valid GAFE addressing.

1. Send the GAFE\_DAC1\_Write command to the GAFE being tested with a data field of 16'h00 (0).
2. Send the GAFE\_DAC1\_Read command. The data field in the return data stream should be 16'h00 (0).
3. Send the GAFE\_DAC1\_Write command to the GAFE being tested with a data field of 16'h3F (63).
4. Send the GAFE\_DAC1\_Read command. The data field in the return data stream should be 16'h3F (63).
5. Send the GAFE\_DAC1\_Write command to the GAFE being tested with a data field of 16'h39 (57).
6. Send the GAFE\_DAC1\_Read command. The data field in the return data stream should be 16'h39 (57).

### **6.36 GAFE ASIC VETO VERNIER Register Test**

This section tests the proper functioning of the GAFE DAC #2 register. Repeat for all GAFE ASICs present, using valid GAFE addressing.

1. Send the GAFE\_DAC2\_Write command to the GAFE being tested with a data field of 16'h00 (0).
2. Send the GAFE\_DAC2\_Read command. The data field in the return data stream should be 16'h00 (0).
3. Send the GAFE\_DAC2\_Write command to the GAFE being tested with a data field of 16'h3F (63).
4. Send the GAFE\_DAC2\_Read command. The data field in the return data stream should be 16'h3F (63).
5. Send the GAFE\_DAC2\_Write command to the GAFE being tested with a data field of 16'h26 (38).
6. Send the GAFE\_DAC2\_Read command. The data field in the return data stream should be 16'h26 (38).

### **6.37 GAFE ASIC HLD Register Test**

This section tests the proper functioning of the GAFE DAC #3 register. Repeat for all GAFE ASICs present, using valid GAFE addressing.

1. Send the GAFE\_DAC3\_Write command to the GAFE being tested with a data field of 16'h00 (0).

2. Send the GAFE\_DAC3\_Read command. The data field in the return data stream should be 16'h00 (0).
3. Send the GAFE\_DAC3\_Write command to the GAFE being tested with a data field of 16'h3F (63).
4. Send the GAFE\_DAC3\_Read command. The data field in the return data stream should be 16'h3F (63).
5. Send the GAFE\_DAC3\_Write command to the GAFE being tested with a data field of 16'h37 (55).
6. Send the GAFE\_DAC3\_Read command. The data field in the return data stream should be 16'h37 (55).

### **6.38 GAFE ASIC BIAS Register Test**

This section tests the proper functioning of the GAFE DAC #4 register. Repeat for all GAFE ASICs present, using valid GAFE addressing.

1. Send the GAFE\_DAC4\_Write command to the GAFE being tested with a data field of 16'h00 (0).
2. Send the GAFE\_DAC4\_Read command. The data field in the return data stream should be 16'h00 (0).
3. Send the GAFE\_DAC4\_Write command to the GAFE being tested with a data field of 16'h3F (63).
4. Send the GAFE\_DAC4\_Read command. The data field in the return data stream should be 16'h3F (63).
5. Send the GAFE\_DAC4\_Write command to the GAFE being tested with a data field of 16'h20 (32).
6. Send the GAFE\_DAC4\_Read command. The data field in the return data stream should be 16'h20 (32).

### **6.39 GAFE ASIC TCI Register Test**

This section tests the proper functioning of the GAFE DAC #5 register. Repeat for all GAFE ASICs present, using valid GAFE addressing.

1. Send the GAFE\_DAC5\_Write command to the GAFE being tested with a data field of 16'h00 (0).
2. Send the GAFE\_DAC5\_Read command. The data field in the return data stream should be 16'h00 (0).
3. Send the GAFE\_DAC5\_Write command to the GAFE being tested with a data field of 16'h3F (63).
4. Send the GAFE\_DAC5\_Read command. The data field in the return data stream should be 16'h3F (63).
5. Send the GAFE\_DAC5\_Write command to the GAFE being tested with a data field of 16'h00 (0).
6. Send the GAFE\_DAC5\_Read command. The data field in the return data stream should be 16'h00 (0).

At the successful conclusion of this section, the GARC commandable register bits have been demonstrated to be functional with all bits toggling as commanded.

**6.40 Test of the GARC Parity Error Detection and Error Generation**

This section tests the proper functioning of the GARC return data parity select bit and the GARC's ability to detect command and data parity errors.

This test may be automated using the LabView GSE via the **GARC Parity Test.txt** script.

1. Send the GARC\_Reset command.
2. Send the GARC\_Status command. The data field in the return data stream should be decimal 24.
3. Send the GARC\_Mode command with a data argument of decimal 769, a command to return event data with even parity.
4. Send the GARC\_Status command. The data field in the return data stream should show an AEM parity error.
5. Send the GARC\_Mode command with a data argument of decimal 768, a command back to odd parity, the nominal mode.
6. Send the GARC\_Status command. The data field in the return data stream should be decimal 24. The AEM should indicate nominal parity (no errors).
7. Send the GARC\_Reset command to reinitialize all GARC registers. Ready to test Command and Data parity errors.
8. Send the GARC\_Diagnostic command and verify that bits 15:12 in the return data word are 4'b0000.
9. Send the GARC\_Version command.
10. Send the GARC\_Diagnostic command and verify that bits 15:12 in the return data word are 4'b0000.
11. Send the Command\_Register command and verify that the return data is 16'h0000.
12. Send the even command parity (i.e., error) command: 34'h2404C0001.
13. Send the GARC\_Diagnostic command and verify that bits 15:12 in the return data word are 4'b1101.
14. Send the Command\_Register command and verify that the return data is 16'h404E.
15. At the successful conclusion of this section, please document this fact by making an entry in the copy of the Comprehensive Performance Test Record associated with this procedure.

**6.41 Test of the GARC Diagnostic Status Register**

This test may be automated using the LabView GSE via the **GARC Diagnostic Status Reg Test.tst** script.

1. Send the GARC\_Reset command to initialize the GARC registers.
2. Send the GARC\_Mode\_Rd command. The data field in the return data stream should be decimal 768.
3. Send the GARC\_Status command. The data field in the return data stream should be decimal 24.
4. Send the GARC\_Cmd\_Reg command. The data field in the return data stream should be decimal 0.
5. Send the GARC\_Diagnostic command. The most significant four bits in the return data stream are the Diagnostic bits. The remaining 12 bits are state machine loop and command counters and the total should not be zero. The data field in the return data stream should be decimal 771.

#### 6.42 Command Counter Test

This test may be automated using the LabView GSE via the GARC Command Counter Test.txt script.

1. Send the GARC\_Reset command to initialize the GARC registers.
2. Send the GARC\_Diagnostic command. The value in bits 7:0 of the returned data word should have the value 0.
3. Send the GARC\_Version command ten times.
4. Send the GARC\_Diagnostic command. The value of the returned data word should have the value 2827.

#### 6.43 Test of the Look-At-Me Circuitry

This section tests the proper functioning of the GARC Look-At-Me circuitry. The GARC has the capability to receive commands from either the primary side or secondary side. The selection of which set of receivers to listen to is controlled by the Look-At-Me circuitry. This circuitry toggles the status of the receiving side based upon receipt of a special command pattern (e.g., **34'h24153D721**). This the equivalent of sending a GARC configuration command to addr 1, function 4, with a data pattern of 60304.

Another way to look at the bit pattern (in a format like the ICD would present it) would be:

1001	==	configuration command
0	==	GARC
00001	==	address 1
0	==	write
1	==	a command with data
0100	==	function 4
1	==	command parity
16'h	==	EB90 == data field (60304)
0	==	data parity

**This test may be automated using the LabView GSE via the GARC Look At Me Test.txt script.**

Turn on the system and send a Look-at-me command. Send the GARC\_Status command. The data field in the return data stream should be decimal 24. Bit 0 (LSB) of the Status register represents the Look-At-Me status (0 = A, 1 = B).. Set the GSE to the Secondary side, Send the GARC\_Status command. Observe that there is no response from the GARC.

Send the Look\_At\_Me command to the GARC from the secondary side interface. Send the GARC\_Status command. The data field in the return data should be decimal 25 (with the LSB = 1, indicating the Look-At-Me status is B side). Set the GSE for the Primary side and send the GARC\_Status command. Observe that there is no response from the GARC. Send the Look\_At\_Me command to the GARC from the primary side interface. Send the GARC\_Status command. The data field in the return data should be decimal 24 (with the LSB = 0, indicating the Look-At-Me status is A side).

This concludes the test of the GARC Look-At-Me circuitry. At the successful conclusion of this section, please document this fact by making an entry in the copy of the Comprehensive Performance Test Record associated with this procedure.

## 7.0 Test of the GARC PHA Functions

This section tests the proper functions and commandability of the event data processor as it handles the PHA Data from the GAFE's

### 7.1 Maximum PHA Return Test

This section tests the proper functioning of the event data processor as it handles the Maximum Number of PHA words command.

This test may be automated using the LabView GSE via the GARC Max PHA Return Test.txt script

1. Send the Max\_PHA\_Wr command with a data argument of 18. Send the Max\_PHA\_Rd command and verify that the data value has been set. Send the Trigger\_ZS command and verify that the event data processor sends back 18 PHA words.
2. Repeat the previous step with a data argument of 17.
3. Repeat the previous step with a data argument of 16.
4. Repeat the previous step with a data argument of 15.
5. Repeat the previous step with a data argument of 14.
6. Repeat the previous step with a data argument of 13.
7. Repeat the previous step with a data argument of 12.
8. Repeat the previous step with a data argument of 11.
9. Repeat the previous step with a data argument of 10.
10. Repeat the previous step with a data argument of 9.
11. Repeat the previous step with a data argument of 8.
12. Repeat the previous step with a data argument of 7.
13. Repeat the previous step with a data argument of 6.
14. Repeat the previous step with a data argument of 5.
15. Repeat the previous step with a data argument of 4.
16. Repeat the previous step with a data argument of 3.
17. Repeat the previous step with a data argument of 2.
18. Repeat the previous step with a data argument of 1.
19. Repeat the previous step with a data argument of 0.
20. Send the GARC\_Reset command to restore the Max\_PHA number to the default. This completes the test of the GARC MAX\_PHA function. At the successful conclusion of this section, please document this fact by making an entry in the copy of the Comprehensive Performance Test Record associated with this procedure.

### 7.2 PHA Enable/Disable Test

This section tests the proper functioning of the event data processor as it handles the PHA enables and disables in selection of PHA words for transmission.

This test may be automated using the LabView GSE via the **GARC PHA Enable Test.txt** script.

1. Send the PHA\_En0\_Wr command with a data argument of decimal 'hFFFF (65535). Verify this value with the PHA\_En0\_Rd command. This enables channels 0 – 15.
2. Send the PHA\_En1\_Wr command with a data argument of decimal 3. Verify this value with the PHA\_En1\_Rd command. This enables channels 16 and 17.
3. Send the Trigger\_NOZS command. The event data processor should respond with 18 PHA words during event readout.
4. Send the PHA\_En0\_Wr command with a data argument of 'hFFFE (65534). Verify this value with the PHA\_En0\_Rd command. Send the Trigger\_NOZS command. The event processor should respond with all PHA words except channel 0.
5. Send the PHA\_En0\_Wr command with a data argument of 'hFFFD (65533). Verify this value with the PHA\_En0\_Rd command. Send the Trigger\_NOZS command. The event processor should respond with all PHA words except channel 1.
6. Send the PHA\_En0\_Wr command with a data argument of 'hFFFB (65531). Verify this value with the PHA\_En0\_Rd command. Send the Trigger\_NOZS command. The event processor should respond with all PHA words except channel 2.
7. Send the PHA\_En0\_Wr command with a data argument of 'hFFF7 (65527). Verify this value with the PHA\_En0\_Rd command. Send the Trigger\_NOZS command. The event processor should respond with all PHA words except channel 3.
8. Send the PHA\_En0\_Wr command with a data argument of 'hFFE7 (65519). Verify this value with the PHA\_En0\_Rd command. Send the Trigger\_NOZS command. The event processor should respond with all PHA words except channel 4.
9. Send the PHA\_En0\_Wr command with a data argument of 'hFFDF (65503). Verify this value with the PHA\_En0\_Rd command. Send the Trigger\_NOZS command. The event processor should respond with all PHA words except channel 5.
10. Send the PHA\_En0\_Wr command with a data argument of 'hFFBF (65471). Verify this value with the PHA\_En0\_Rd command. Send the Trigger\_NOZS command. The event processor should respond with all PHA words except channel 6.
11. Send the PHA\_En0\_Wr command with a data argument of 'hFF7F (65407). Verify this value with the PHA\_En0\_Rd command. Send the Trigger\_NOZS command. The event processor should respond with all PHA words except channel 7.
12. Send the PHA\_En0\_Wr command with a data argument of 'hFEFF (65279). Verify this value with the PHA\_En0\_Rd command. Send the Trigger\_NOZS command. The event processor should respond with all PHA words except channel 8.
13. Send the PHA\_En0\_Wr command with a data argument of 'hFDFF (65023). Verify this value with the PHA\_En0\_Rd command. Send the Trigger\_NOZS command. The event processor should respond with all PHA words except channel 9.
14. Send the PHA\_En0\_Wr command with a data argument of 'hFBFF (64511). Verify this value with the PHA\_En0\_Rd command. Send the Trigger\_NOZS command. The event processor should respond with all PHA words except channel 10.

15. Send the PHA\_En0\_Wr command with a data argument of 'hF7FF (63487). Verify this value with the PHA\_En0\_Rd command. Send the Trigger\_NOZS command. The event processor should respond with all PHA words except channel 11.
16. Send the PHA\_En0\_Wr command with a data argument of 'hEFFF (61439). Verify this value with the PHA\_En0\_Rd command. Send the Trigger\_NOZS command. The event processor should respond with all PHA words except channel 12.
17. Send the PHA\_En0\_Wr command with a data argument of 'hDFFF (57343). Verify this value with the PHA\_En0\_Rd command. Send the Trigger\_NOZS command. The event processor should respond with all PHA words except channel 13.
18. Send the PHA\_En0\_Wr command with a data argument of 'hBFFF (49151). Verify this value with the PHA\_En0\_Rd command. Send the Trigger\_NOZS command. The event processor should respond with all PHA words except channel 14.
19. Send the PHA\_En0\_Wr command with a data argument of 'h7FFF (32767). Verify this value with the PHA\_En0\_Rd command. Send the Trigger\_NOZS command. The event processor should respond with all PHA words except channel 15.
20. Send the PHA\_En0\_Wr command with a data argument of 'hFFFF (65535). Verify this value with the PHA\_En0\_Rd command. Send the Trigger\_NOZS command. The event processor should respond with all PHA words.
21. Send the PHA\_En1\_Wr command with a data argument of 2. Verify this value with the PHA\_En1\_Rd command. Send the Trigger\_NOZS command. The event processor should respond with all PHA words except channel 16.
22. Send the PHA\_En1\_Wr command with a data argument of 1. Verify this value with the PHA\_En1\_Rd command. Send the Trigger\_NOZS command. The event processor should respond with all PHA words except channel 17.
23. Send the GARC\_Reset command to reinitialize all GARC registers. This completes the testing of the PHA enable and disable register bits. At the successful conclusion of this section, please document this fact by making an entry in the copy of the Comprehensive Performance Test Record associated with this procedure.

### 7.3 PHA Threshold Verification Test

This section tests the proper functioning of each of the 18 PHA thresholding circuits. The intent of the circuits is such that the ZS Map bit is set active high for a given PHA word if either the range bit is high or the 12 bit PHA word is greater than the PHA threshold.

This test may be automated using the LabView GSE via the **GARC Chip PHA Threshold Test.txt** script.

1. Send the GARC\_Reset command.
2. Send the Max\_PHA\_Wr command with a data argument of 18.
3. Send the PHA\_Thresh00\_Wr command with a data argument of 100.
4. Repeat these steps for channels 01 through 17.
5. Send the TRIG\_ZS command. All 18 channels of PHA should be returned.
6. Send the PHA\_Thresh00\_Wr command with a data argument of 2000. Verify with the PHA\_Thresh00\_Rd command.
7. Send the TRIG\_ZS command. All PHA channels except for channel 0 should be returned in the event data.
8. Send the PHA\_Thresh01\_Wr command with a data argument of 3000 and set PHA\_Thresh00 back to 100. Verify with the PHA\_ThreshXX\_Rd command.

9. Send the TRIG\_ZS command. All PHA channels except for channel 1 should be returned in the event data.
10. Repeat the above two steps serially for channels 2 – 17, verifying in each case that the proper event word is zero suppressed in the event data.
11. Send the GARC\_Reset command to restore all values to the power-on defaults.

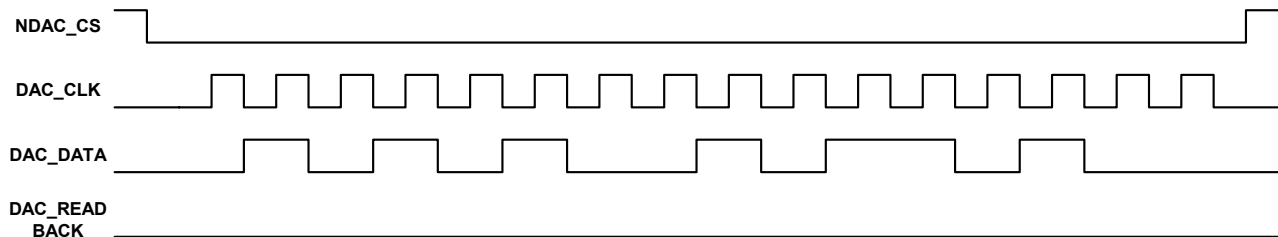
This completes the PHA Threshold Verification section of the procedure. Record the successful completion of this test in the Test Record at the end of this procedure.

## 8.0 Test of the MAX5121 DAC and Associated High Voltage Supply Control Functions

This section tests the proper commanding of the MAX5121 DAC and verify's the High Voltage Supply interface controls.

### 8.1 Capture of the DAC Control Signals

1. Connect the probe on channel 1 to NDAC\_CS. Set the scope to trigger on the negedge of this signal. This is J7-4 on the GARC test board.
2. Connect the probe on channel 2 to DAC\_CLK. (J7-1)
3. Connect the probe on channel 3 to DAC\_DATA. (J7-2)
4. Connect the probe on channel 4 to DAC\_READBACK. (J7-3)
5. Send the HVBS\_Level\_Wr command with a data argument of hex A5A (decimal 2650). Send the HVBS\_Level\_Rd command to verify.
6. Check that the scope is armed and then send the Use\_HV\_Nominal\_Wr command. The scope should trigger and the following waveform should be displayed. Send the command twice the DAC\_Read Back signal should be the same as the DAC\_Data except it will be delayed one clock cycle. Verify that the DAC clock is nominally 5 MHz \_\_\_\_\_ (check)
- 7.



DAC DATA WRITE WITH DATA PATTERN HEX A5A

### DAVE – SHOW THE DAC READBACK PULSES AFTER THE SECOND WR COMMAND

8. Check that the scope is armed and then send the Use\_HV\_Nominal\_Rd command. The software readback should capture the following value in the 16 bit return word: hex F4B4 (decimal 31348). This value may also be seen on the DAC DATA READBACK scope trace.
9. Move the probe on channel 1 to NDAC\_CLR (J7-5 of the GARC Test Board). Verify the scope is armed and send the GARC\_Reset command. The scope should trigger and a valid active low clear signal should be observed on the scope.
  - a. Duration of the RESET pulse: \_\_\_\_\_ (expected ~ 150 ns)

### 8.2 Test of the GARC SAA/HV Normal Modes

This section tests the proper functioning of the circuitry that commands the MAX5121 DAC.

1. Send the HVBS\_Level\_Wr command with a data argument of decimal 2048. Send the HVBS\_Level\_Rd command to verify the write command.
2. Send the SAA\_Level\_Wr command with a data argument of decimal 1024. Send the SAA\_Level\_Rd command to verify.

3. Using a multimeter on the DAC voltage reference, verify that the MAX5121 reference is at 1.25V. This is J5-3 on the GARC test board. **Value observed:** \_\_\_\_\_
4. Using a multimeter on the DAC output, verify that the MAX5121 output is at 0V. This is J5-1 on the GARC test board. **Value observed:** \_\_\_\_\_
- 5.
6. Send the Use\_HV\_Nominal command. Verify that the MAX5121 output goes to half scale, e.g., 0.625 V at J5-1. **Value observed:** \_\_\_\_\_
7. Send the DAC\_HVReg\_Rd command and verify the return data pattern from the MAX5121 DAC is decimal 10240.
8. Send the Use\_SAA\_Level command. Verify that the MAX5121 output goes to 1/4 scale, e.g., 0.312V at J5-1. **Value observed:** \_\_\_\_\_
9. Send the DAC\_SAAREg\_Rd command and verify the return data pattern from the MAX5121 DAC is decimal 9216.
10. Send the GARC\_Reset command. Verify that the MAX5121 output goes to 0 V.

### 8.3 Test of the HVBS Triple Modular Redundancy Circuitry

This section tests the proper functioning of the GARC HVBS enable circuitry. Each pattern in the TMR logic will be tested to verify proper recovery from a single event upset. The GARC HV\_ENABLE\_1 pin is 185 (J7-6) and the HV\_ENABLE\_2 pins is 186 (J7-7). A truth table for proper TMR circuitry function is detailed below. This test may be automated using the LabView GSE via the **GARC HV Enable Test.txt** script.

Enable Bit A	Enable Bit B	Enable Bit C	HV Enable Output
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

HV\_Enable\_1 is on J7-6 and HV\_Enable\_2 is on J7-7 on the GARC test board.

1. Send the GARC\_Mode\_Wr command with a data argument of 768 and verify the data with the GARC\_Mode\_Rd command.
2. Send the GARC\_Status command. The data field in the return data stream should show that bits 1 and 2 are 0, indicating that HVBS 1 and 2, respectively, are disabled. The data field value should be 24. Verify that GARC pins 185 and 186 are at 0V.
3. Send the GARC\_Mode\_Wr command with decimal argument 770 and verify the data with the GARC\_Mode\_Rd command. Send the GARC\_Status command and verify that HVBS 1 is disabled and HVBS 2 is disabled. The data field value should be 24. Verify that GARC pins 185 and 186 are at 0V.

4. Send the GARC\_Mode\_Wr command with decimal argument 772 and verify the data with the GARC\_Mode\_Rd command. Send the GARC\_Status command and verify that HVBS 1 is disabled and HVBS 2 is disabled. The data field value should be 24. Verify that GARC pins 185 and 186 are at 0V.
5. Send the GARC\_Mode\_Wr command with decimal argument 774 and verify the data with the GARC\_Mode\_Rd command. Send the GARC\_Status command and verify that HVBS 1 is enabled and HVBS 2 is disabled. . The data field value should be 26. Verify that GARC pins 185 is at 3.3V and pin 186 is at 0V.
6. Send the GARC\_Mode\_Wr command with decimal argument 776 and verify the data with the GARC\_Mode\_Rd command. Send the GARC\_Status command and verify that HVBS 1 is disabled and HVBS 2 is disabled. . The data field value should be 24. Verify that GARC pins 185 is at 0V and pin 186 is at 0V.
7. Send the GARC\_Mode\_Wr command with decimal argument 778 and verify the data with the GARC\_Mode\_Rd command. Send the GARC\_Status command and verify that HVBS 1 is enabled and HVBS 2 is disabled. . The data field value should be 26. Verify that GARC pins 185 is at 3.3V and pin 186 is at 0V.
8. Send the GARC\_Mode\_Wr command with decimal argument 780 and verify the data with the GARC\_Mode\_Rd command. Send the GARC\_Status command and verify that HVBS 1 is enabled and HVBS 2 is disabled. The data field value should be 26. Verify that GARC pins 185 is at 3.3V and pin 186 is at 0V.
9. Send the GARC\_Mode\_Wr command with decimal argument 782 and verify the data with the GARC\_Mode\_Rd command. Send the GARC\_Status command and verify that HVBS 1 is enabled and HVBS 2 is disabled. The data field value should be 26. Verify that GARC pins 185 is at 3.3V and pin 186 is at 0V.
10. Send the GARC\_Mode\_Wr command with decimal argument 784 and verify the data with the GARC\_Mode\_Rd command. Send the GARC\_Status command and verify that HVBS 1 is disabled and HVBS 2 is disabled. The data field value should be 24. Verify that GARC pins 185 is at 0V and pin 186 is at 0V.
11. Send the GARC\_Mode\_Wr command with decimal argument 800 and verify the data with the GARC\_Mode\_Rd command. Send the GARC\_Status command and verify that HVBS 1 is disabled and HVBS 2 is disabled. The data field value should be 24. Verify that GARC pins 185 is at 0V and pin 186 is at 0V.
12. Send the GARC\_Mode\_Wr command with decimal argument 816 and verify the data with the GARC\_Mode\_Rd command. Send the GARC\_Status command and verify that HVBS 1 is disabled and HVBS 2 is enabled. The data field value should be 28. Verify that GARC pins 185 is at 0V and pin 186 is at 3.3V.
13. Send the GARC\_Mode\_Wr command with decimal argument 832 and verify the data with the GARC\_Mode\_Rd command. Send the GARC\_Status command and verify that HVBS 1 is disabled and HVBS 2 is disabled. The data field value should be 24. Verify that GARC pins 185 is at 0V and pin 186 is at 0V.
14. Send the GARC\_Mode\_Wr command with decimal argument 848 and verify the data with the GARC\_Mode\_Rd command. Send the GARC\_Status command and verify that HVBS 1 is disabled and HVBS 2 is enabled. The data field value should be 28. Verify that GARC pins 185 is at 0V and pin 186 is at 3.3V.
15. Send the GARC\_Mode\_Wr command with decimal argument 864 and verify the data with the GARC\_Mode\_Rd command. Send the GARC\_Status command and verify that HVBS 1 is disabled

- and HVBS 2 is enabled. The data field value should be 28. Verify that GARC pins 185 is at 0V and pin 186 is at 3.3V.
16. Send the GARC\_Mode\_Wr command with decimal argument 880 and verify the data with the GARC\_Mode\_Rd command. Send the GARC\_Status command and verify that HVBS 1 is disabled and HVBS 2 is enabled. The data field value should be 28. Verify that GARC pins 185 is at 0V and pin 186 is at 3.3V.
  17. Send the GARC\_Mode\_Wr command with decimal argument xxx and verify the data with the GARC\_Mode\_Rd command. Measure the voltage at the two HV\_ENABLE pins:
    - a. HV\_ENABLE\_1 voltage: \_\_\_\_\_ (expected ~ 3.2V)
    - b. HV\_ENABLE\_2 voltage: \_\_\_\_\_ (expected ~ 3.2V)
  18. Send the GARC\_Mode\_Wr command with decimal argument 768 and verify the data with the GARC\_Mode\_Rd command. Measure the voltage at the two HV\_ENABLE pins:
    - a. HV\_ENABLE\_1 voltage: \_\_\_\_\_ (expected ~ 0.015V)
    - b. HV\_ENABLE\_2 voltage: \_\_\_\_\_ (expected ~ 0.015V)
  19. Send the GARC\_Reset command. Verify that GARC pins 185 is at 0V and pin 186 is at 0V.

This completes the test of the HVBS enable/disable TMR circuitry.

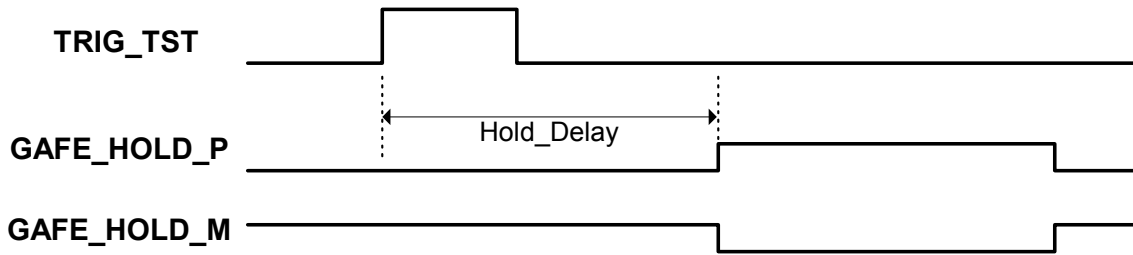
## 9.0 GARC Test Pin Mux Verification

This section tests the proper functionality, switch the GARC test pin between the “Live” and “HitMap Test” outputs. The discriminator signal from the pulse generator is added to the GAFE simulator board at J3-3.

1. Send the GARC\_Mode\_Wr command with a data argument of 1792 to switch the GARC test pin mux to view the “live” signal. Send the GARC\_Mode\_Rd command to verify the register data. When configuration commands are sent, the “live” signal should go inactive low during the state machine busy time. This may be verified via the oscilloscope. The GARC test pin is pin 179 (J7-8 on the GARC test board).
2. Send the GARC\_Mode\_Wr command with a data argument of 768 to switch the GARC test pin mux to view the “HitMap test” signal. Send the GARC\_Mode\_Rd command to verify the register data. When trigger commands are sent, the “HitMap test” signal should go active high during times when there is a delayed VETO signal present at the DISC\_IN input. This may be verified via the oscilloscope. The GARC test pin is pin 179 (J7-8 on the GARC test board).
3. Send the GARC\_Reset command to ensure all registers are at the proper initial value.

## 10.0 Test of the Hold Delay Operation

This section tests the proper functioning of the adjustment of the commandable delay function of the trigger to shaping amplifier hold signal. The trigger signal may be monitored on the GARC at pin 180, TRIG\_TST (GARC test board J7-9). The Hold signal may be monitored differentially on the GARC, GAFE\_HOLD\_P at pin 160 and GAFE\_HOLD\_M at pin 161, as shown in the diagram below. GAFE\_HOLD may be measured across R83 on the GAFE simulator board.



### TRIGGER-TO-HOLD MEASUREMENT

1. Measure the DC levels of both GAFE\_HOLD\_P and GAFE\_HOLD\_M and record the values below:
  - a. GAFE\_HOLD\_P (DC) : \_\_\_\_\_ (expected ~ 870 mV)
  - b. GAFE\_HOLD\_M (DC): \_\_\_\_\_ (expected ~1600 mV)
2. Send the Hold\_Delay\_Wr command with a data argument of decimal 0 and verify with the Hold\_Delay\_Rd command. The Hold\_Delay is variable with arguments of 0 – 127, representing 50 – 6400 ns. Hold can be measured at TP4 on the GAFE simulator. The LabView test script **GARC Hold Delay Test.txt** may be used to partially automate this test. Trigger on J7-9 on the GARC test board “TRIG\_TST”.

Hold Delay Command	Measured Hold Delay (ns)	Expected Hold Delay (ns)
0		50
1		100
2		150
3		200
4		250
5		300
6		350
7		400
8		450
9		500
10		550
11		600
12		650
13		700
14		750
15		800
16		850
17		900
18		950
19		1000
20		1050
21		1100
22		1150
23		1200
24		1250
25		1300
26		1350
27		1400

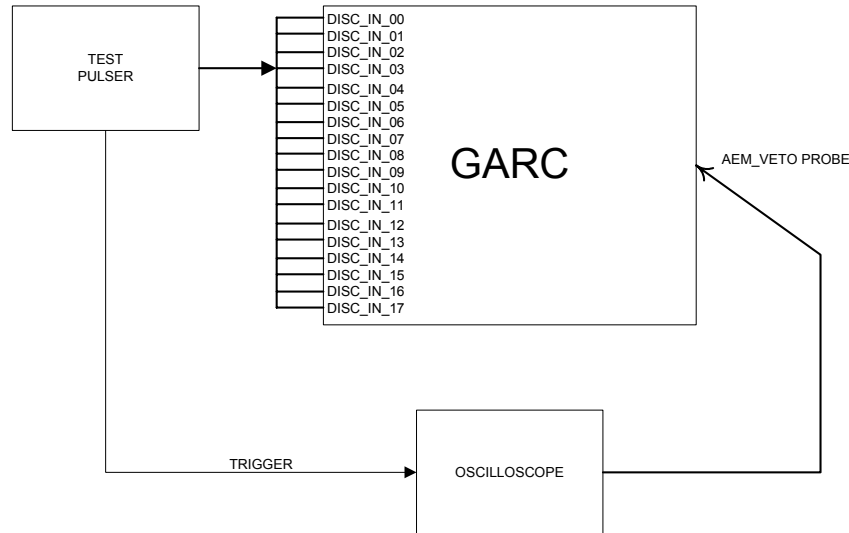
28		1450
29		1500
30		1550
31		1600
32		1650
33		1700
34		1750
35		1800
36		1850
37		1900
38		1950
39		2000
40		2050
41		2100
42		2150
43		2200
44		2250
45		2300
46		2350
47		2400
48		2450
49		2500
50		2550
51		2600
52		2650
53		2700
54		2750
55		2800
56		2850
57		2900
58		2950
59		3000
60		3050
61		3100
62		3150
63		3200
64		3250
65		3300
66		3350
67		3400
68		3450
69		3500
70		3550
71		3600
72		3650
73		3700
74		3750
75		3800
76		3850
77		3900
78		3950
79		4000
80		4050
81		4100
82		4150
83		4200

84		4250
85		4300
86		4350
87		4400
88		4450
89		4500
90		4550
91		4600
92		4650
93		4700
94		4750
95		4800
96		4850
97		4900
98		4950
99		5000
100		5050
101		5100
102		5150
103		5200
104		5250
105		5300
106		5350
107		5400
108		5450
109		5500
110		5550
111		5600
112		5650
113		5700
114		5750
115		5800
116		5850
117		5900
118		5950
119		6000
120		6050
121		6100
122		6150
123		6200
124		6250
125		6300
126		6350
127		6400

3. Send the HitMap\_Width\_Wr command with a data argument of 0. Verify the command using the HitMap\_Width\_Rd readback.
4. Send the GARC\_Reset command to restore all register values to their initial state.

### 11.0 Test of the AEM VETO Signal Functionality

This section tests the proper functioning of the commandable functions of the AEM\_VETO signals. This test requires the use of an oscilloscope, a test pulser for input stimulus, and test points to monitor the AEM\_VETOs. The test setup is as shown below.

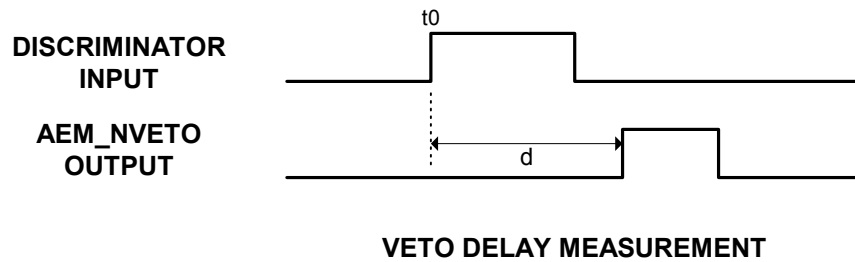


**GARC VETO Test Setup**

All 18 of the discriminator inputs are tied together and also to the output of the test pulser for this portion of the test. The input test pulse should be approximately 200 ns wide and the rate should be set at approximately 20 kHz. The oscilloscope probe will be monitoring the 18 AEM\_VETO outputs. Between commands, this probe will be moved manually from test point to test point to perform this test. Setting the scope to acquire on the pulse output, this test will be measuring the delay (time from trigger to VETO leading edge) and width (time between leading and trailing edges) of each of the VETO pulses. Since each input has the same signal, the outputs should also be common in delay and width. In timing measurements, the activation signal from the test pulser will be the relative zero.

#### 11.1 Veto Delay Test

1. The discriminator input signal enters the GAFE simulator at J3-3. The NVETO output signal is monitored at J8-2 on the AEM simulator GSE board. For this measurement, it may be convenient to use the oscilloscope in the “persist” mode and then monitor the minimum delay. This test may be automated using the LabView GSE via the script **GARC VETO Delay Test.txt**.
2. Send the Veto\_Delay\_Wr command with a data argument of 0. Confirm this command via the Veto\_Delay\_Rd command.
3. Send the Veto\_Width\_Wr command with a data argument of 0. Confirm this command via the Veto\_Width\_Rd command. Using the oscilloscope, measure the delay from scope trigger to leading edge of VETO for each of the 18 pulses and record the data in the table below. Note that all 18 pulses are designed to have identical timing for this test. The VETO Delay is the time **d** in the diagram below.



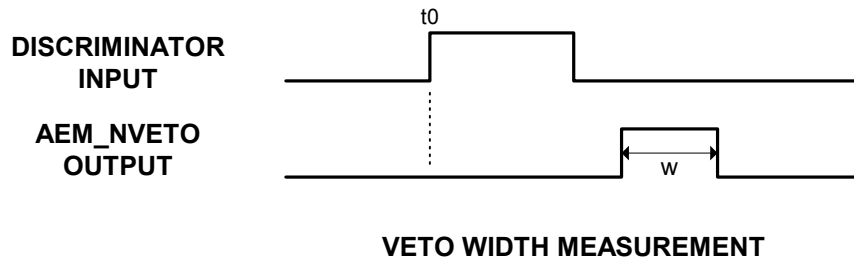
4. Repeat the Veto\_Delay\_Wr, Veto\_Delay\_Rd sequence for data argument values of decimal 1 to 31. Record the results in the table below. Note that there is a 50 ns jitter on each of the “expected” numbers.

Veto_Delay Setting	Delay from Scope trigger to Leading Edge (ns)	Expected (ns)
0		200
1		250
2		300
3		350
4		400
5		450
6		500
7		550
8		600
9		650
10		700
11		750
12		800
13		850
14		900
15		950
16		1000
17		1050
18		1100
19		1150
20		1200
21		1250
22		1300
23		1350
24		1400
25		1450
26		1500
27		1550
28		1600
29		1650
30		1700
31		1750

5. Send the Veto\_Delay\_Wr command with a data argument of 0 and a Veto\_Delay\_Rd command to verify this value.

### 11.2 Veto Width Test

1. The setup for this test is the same as the VETO Delay test above. This test may be automated using the LabView GSE via the **GARC VETO Width Test.txt** script.
2. Send the Veto\_Width\_Wr command with a data argument of 0. Confirm this command via the Veto\_Width\_Rd command. Using the oscilloscope, measure the width of each of the 18 VETO pulses and record the data in the table below. Note that the width of each of these pulses is expected to be identical for any given commanded setting. The VETO Width is the “w” parameter in the diagram below.



3. Repeat the Veto\_Width\_Wr, Veto\_Width\_Rd sequence for data argument values of decimal 1 to 7. Record the results in the table below.

Veto_Width	Measured Width of VETO Pulse (ns)	Expected Width (ns)
0		50
1		100
2		150
3		200
4		250
5		300
6		350
7		400

### 11.3 Veto Enable and Disable Test

This test may be automated when using the LabView GSE via test script **GARC VETO Disable A/B SideTest.txt**. Monitor the counters to verify the Veto signals.

1. Send the GARC\_Mode\_Wr command with the value of 768 to enable the “A” and enable the “B” side VETOs. Send the GARC\_Mode\_Rd command and verify that the return data has a value of 768. Generate a Trigger event and measure the Veto amplitude swing at R1 and R2 on the AEM Simulator for Channel 12 for both A and B.  
R1 - \_\_\_\_\_ (expected 400mV)  
R2 - \_\_\_\_\_ (expected 400mV)
2. Send the GARC\_Mode\_Wr command with the value of 512 to disable the “A” and enable the “B” side VETOs. Send the GARC\_Mode\_Rd command and verify that the return data has a value of 512. Generate a Trigger event and measure the Veto amplitude swing at R1 and R2 on the AEM Simulator for Channel 12 for both A and B.  
R1 - \_\_\_\_\_ (expected 400mV)

R2 - \_\_\_\_\_(expected 50mV)

3. Send the GARC\_Mode\_Wr command with the value of 256 to enable the “A” and disable the “B” side VETOs. Send the GARC\_Mode\_Rd command and verify that the return data has a value of 256. Generate a Trigger event and measure the Veto amplitude swing at R1 and R2 on the AEM Simulator for Channel 12 for both A and B.

R1 - \_\_\_\_\_(expected 50mV)

R2 - \_\_\_\_\_(expected 400mV)

4. Send the GARC\_Mode\_Wr command with the value of 0 to disable both the “A” and “B” side VETOs. Send the GARC\_Mode\_Rd command and verify that the return data has a value of 0. Generate a Trigger event and measure the Veto amplitude swing at R1 and R2 on the AEM Simulator for Channel 12 for both A and B.

R1 - \_\_\_\_\_(expected 50mV)

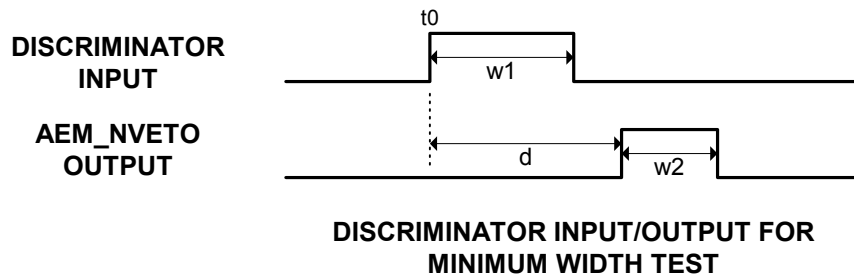
R2 - \_\_\_\_\_(expected 50mV)

5. Send the GARC\_Mode\_Wr command with the value of 256 to enable the “A” and disable the “B” side VETOs. Send the GARC\_Mode\_Rd command and verify that the return data has a value of 256.
6. Send the VETO\_En0\_Rd command and verify that the return data has value ‘hFFFF (65535). Send the VETO\_En1\_Rd command and verify that the return data has the value 3
7. Send the VETO\_En0\_Wr command with a data argument ‘hFFFE (65534). Verify the command with the VETO\_En0\_Rd command. Generate a TCI event for all channels and verify that all VETOs are on with the exception of channel 0.
8. Send the VETO\_En0\_Wr command with a data argument ‘hFFFD (65533). Verify the command with the VETO\_En0\_Rd command. Generate a TCI event for all channels and verify that all VETOs are on with the exception of channel 1.
9. Send the VETO\_En0\_Wr command with a data argument ‘hFFFB (65531). Verify the command with the VETO\_En0\_Rd command. Generate a TCI event for all channels and verify that all VETOs are on with the exception of channel 2.
10. Send the VETO\_En0\_Wr command with a data argument ‘hFFF7 (65527). Verify the command with the VETO\_En0\_Rd command. Generate a TCI event for all channels and verify that all VETOs are on with the exception of channel 3.
11. Send the VETO\_En0\_Wr command with a data argument ‘hFFEF (65519). Verify the command with the VETO\_En0\_Rd command. Generate a TCI event for all channels and verify that all VETOs are on with the exception of channel 4.
12. Send the VETO\_En0\_Wr command with a data argument ‘hFFDF (65503). Verify the command with the VETO\_En0\_Rd command. Generate a TCI event for all channels and verify that all VETOs are on with the exception of channel 5.
13. Send the VETO\_En0\_Wr command with a data argument ‘hFFBF (65471). Verify the command with the VETO\_En0\_Rd command. Generate a TCI event for all channels and verify that all VETOs are on with the exception of channel 6.
14. Send the VETO\_En0\_Wr command with a data argument ‘hFF7F (65407). Verify the command with the VETO\_En0\_Rd command. Generate a TCI event for all channels and verify that all VETOs are on with the exception of channel 7.
15. Send the VETO\_En0\_Wr command with a data argument ‘hFEFF (65279). Verify the command with the VETO\_En0\_Rd command. Generate a TCI event for all channels and verify that all VETOs are on with the exception of channel 8.

16. Send the VETO\_En0\_Wr command with a data argument 'hFDFF (65023). Verify the command with the VETO\_En0\_Rd command. Generate a TCI event for all channels and verify that all VETOs are on with the exception of channel 9.
17. Send the VETO\_En0\_Wr command with a data argument 'hFBFF (64511). Verify the command with the VETO\_En0\_Rd command. Generate a TCI event for all channels and verify that all VETOs are on with the exception of channel 10.
18. Send the VETO\_En0\_Wr command with a data argument 'hF7FF (63487). Verify the command with the VETO\_En0\_Rd command. Generate a TCI event for all channels and verify that all VETOs are on with the exception of channel 11.
19. Send the VETO\_En0\_Wr command with a data argument 'hEFFF (61439). Verify the command with the VETO\_En0\_Rd command. Generate a TCI event for all channels and verify that all VETOs are on with the exception of channel 12.
20. Send the VETO\_En0\_Wr command with a data argument 'hDFFF (57343). Verify the command with the VETO\_En0\_Rd command. Generate a TCI event for all channels and verify that all VETOs are on with the exception of channel 13.
21. Send the VETO\_En0\_Wr command with a data argument 'hBFFF (49151). Verify the command with the VETO\_En0\_Rd command. Generate a TCI event for all channels and verify that all VETOs are on with the exception of channel 14.
22. Send the VETO\_En0\_Wr command with a data argument 'h7FFF (32767). Verify the command with the VETO\_En0\_Rd command. Generate a TCI event for all channels and verify that all VETOs are on with the exception of channel 15.
23. Send the VETO\_En0\_Wr command with a data argument 'hFFFF (65535). Verify the command with the VETO\_En0\_Rd command. Generate a TCI event for all channels and verify that all VETOs are on.
24. Send the VETO\_En1\_Wr command with a data argument 2. Verify the command with the VETO\_En1\_Rd command. Generate a TCI event for all channels and verify that all VETOs are on with the exception of channel 16.
25. Send the VETO\_En1\_Wr command with a data argument 1. Verify the command with the VETO\_En1\_Rd command. Generate a TCI event for all channels and verify that all VETOs are on with the exception of channel 17.
26. Send the GARC\_Mode\_Wr command with the value of 512 to Disable the "A" and enable the "B" side VETOs. Send the GARC\_Mode\_Rd command and verify that the return data has a value of 512. Repeat steps 6 to 25 and verify the B side functionality.
27. Send the GARC\_Reset command to restore all register values to their initial state. At the successful conclusion of this section, please document this fact by making an entry in the copy of the Comprehensive Performance Test Record associated with this procedure.

#### **11.4 Discriminator Input Minimum Width Test**

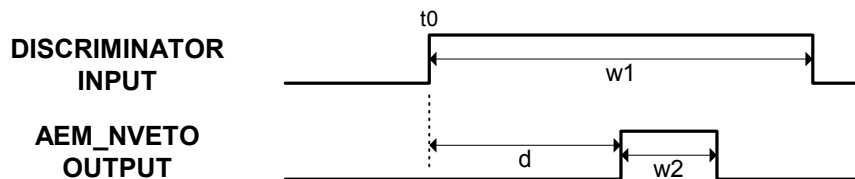
This test verifies that the VETO processing circuitry meets the specifications detailed in the ICD for the input width. For this test, all discriminator inputs greater than 100 ns in width should be processed, those in the 50 ns to 100 ns range should be sometimes processed (due to jitter), and those inputs less than 50 ns should never be processed. The discriminator signal is input to the GAFE simulator board on J3-3. The VETO signals are monitored after the receivers on the AEM Simulator GSE board.



1. Send the VETO\_Delay\_Wr command with a data argument of 0. Verify the command with the VETO\_Delay\_Rd command. This fixes the delay ( $d$  in the figure) at 150 ns for this test.
2. Send the VETO\_Width\_Wr command with a data argument of 0. Verify the command with the VETO\_Width\_Rd command. This fixes the output width ( $w2$  in the figure) at 50 ns for this test.
3. Set the discriminator input on channel 00 to be a pulse of greater than 300 ns in width ( $w1$  in the figure). Verify the ACD\_NVETO\_00 signal out has a pulse output of 50 ns width.
4. Steadily decrease the width ( $w1$ ) of the input pulse down to just greater than 100 ns in width. Verify that no pulses are missed. Adjust the width ( $w1$ ) of the input pulse to between 50 ns and 100 ns in width. Verify that some pulses are there and some are missed due to the jitter in synchronization of the input signal. Adjust the input pulse width ( $w1$ ) to just less than 50 ns and verify that all input pulses are rejected (i.e., no pulses output at NVETO).
5. Repeat this test for channel 01 and verify.
6. Repeat this test for channel 02 and verify.
7. Repeat this test for channel 03 and verify.
8. Repeat this test for channel 04 and verify.
9. Repeat this test for channel 05 and verify.
10. Repeat this test for channel 06 and verify.
11. Repeat this test for channel 07 and verify.
12. Repeat this test for channel 08 and verify.
13. Repeat this test for channel 09 and verify.
14. Repeat this test for channel 10 and verify.
15. Repeat this test for channel 11 and verify.
16. Repeat this test for channel 12 and verify.
17. Repeat this test for channel 13 and verify.
18. Repeat this test for channel 14 and verify.
19. Repeat this test for channel 15 and verify.
20. Repeat this test for channel 16 and verify.
21. Repeat this test for channel 17 and verify.
22. Send the GARC\_Reset command to restore all register values to their initial state.

### 11.5 Discriminator Input Extended Width Test

This test verifies that the VETO processing circuitry meets the specifications detailed in the ICD for extended input widths. For this test, discriminator inputs greater than

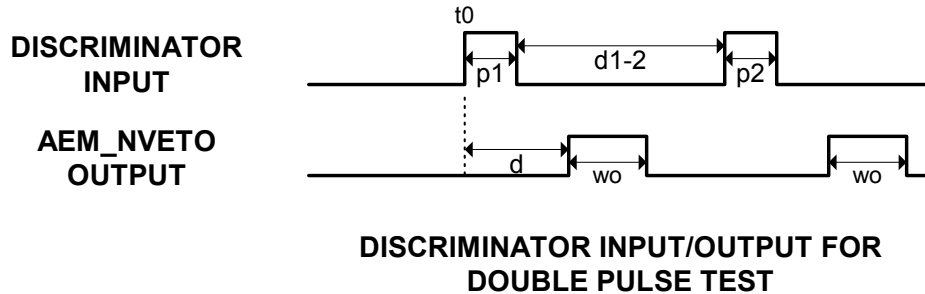


**DISCRIMINATOR INPUT/OUTPUT FOR  
EXTENDED WIDTH TEST**

1. Send the Veto\_Delay\_Wr command with a data argument of 0. Verify with the Veto\_Delay\_Rd command.
2. Send the Veto\_Width\_Wr command with a data argument of 1. Verify with the Veto\_Width\_Rd command.
3. Connect the oscilloscope to monitor the discriminator input and the AEM\_NVETO\_00 output as shown in the diagram above. Start with the width of the input discriminator at approximately 150 ns. Verify that the Veto\_Delay is 150 ns and the Veto\_Width is 100 ns.
4. Increase the discriminator input from 150 ns to 10  $\mu$ s. Note that the output from the AEM\_NVETO\_00 signal does not change.
5. Repeat this test for channel 01 and verify.
6. Repeat this test for channel 02 and verify.
7. Repeat this test for channel 03 and verify.
8. Repeat this test for channel 04 and verify.
9. Repeat this test for channel 05 and verify.
10. Repeat this test for channel 06 and verify.
11. Repeat this test for channel 07 and verify.
12. Repeat this test for channel 08 and verify.
13. Repeat this test for channel 09 and verify.
14. Repeat this test for channel 10 and verify.
15. Repeat this test for channel 11 and verify.
16. Repeat this test for channel 12 and verify.
17. Repeat this test for channel 13 and verify.
18. Repeat this test for channel 14 and verify.
19. Repeat this test for channel 15 and verify.
20. Repeat this test for channel 16 and verify.
21. Repeat this test for channel 17 and verify.
22. Send the GARC\_Reset command to restore all register values to their initial state.

### 11.6 Discriminator Input Double Pulse Test

This test verifies that the VETO processing circuitry meets the specifications detailed in the ICD for extended input widths. For this test, multiple discriminator inputs that cause the VETO outputs to pile-up are verified. The input is on the GAFE simulator board at J3-3 and the outputs are on the AEM Simulator GSE board.

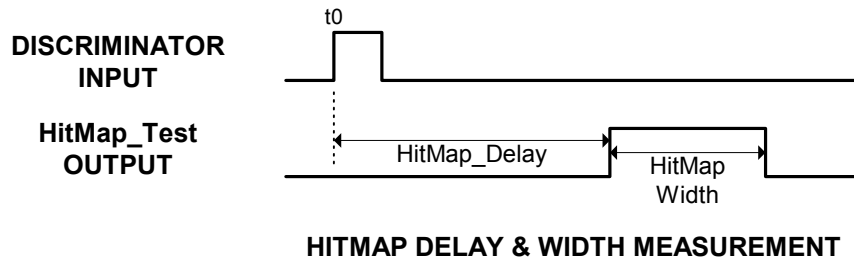


1. Send the Veto\_Delay\_Wr command with a data argument of 0. Verify with the Veto\_Delay\_Rd command.
2. Send the Veto\_Width\_Wr command with a data argument of 3. Verify with the Veto\_Width\_Rd command.
3. Using the test pulser in the double pulse mode, start with the width of the input pulses at 100 ns and the distance, d1-2, at 1  $\mu$ sec. With two pulses input, there should be two pulses output, each 200 ns in width (**wo**) at the AEM\_NVETO\_00 output.
4. Move the input pulses closer until the two AEM\_NVETO\_00 pulses touch. At this point, they should become one extended pulse. As the input pulses move closer, the tail of the extended pulse should become shorter. As the input pulses are separated again by greater than 200 ns, the AEM\_NVETO\_00 should again show two pulses.
5. Repeat this test for channel 01 and verify.
6. Repeat this test for channel 02 and verify.
7. Repeat this test for channel 03 and verify.
8. Repeat this test for channel 04 and verify.
9. Repeat this test for channel 05 and verify.
10. Repeat this test for channel 06 and verify.
11. Repeat this test for channel 07 and verify.
12. Repeat this test for channel 08 and verify.
13. Repeat this test for channel 09 and verify.
14. Repeat this test for channel 10 and verify.
15. Repeat this test for channel 11 and verify.
16. Repeat this test for channel 12 and verify.
17. Repeat this test for channel 13 and verify.
18. Repeat this test for channel 14 and verify.
19. Repeat this test for channel 15 and verify.
20. Repeat this test for channel 16 and verify.

21. Repeat this test for channel 17 and verify.
22. Send the GARC\_Reset command to restore all register values to their initial state.

## 12.0 Test of the HitMap Functionality

This section tests the proper functioning of the commandable functions of the HitMap test point. This test requires the use of an oscilloscope, a test pulser for input stimulus, and test points to monitor the AEM\_VETOs. The test setup is identical to the VETO test setup shown above. Only channel 00 is available on the test pin for HitMap verification. The diagram below details the measurement points to use with the digitizing oscilloscope. The discriminator input signal should be input at the GAFE simulator test connector, J3-3, and should be ~ 100 nsec in width. The HitMap output is at J7-8 on the GARC test board.



### 12.1 HitMap Width Test

This test may be automated using the LabView GSE via the **GARC HitMap Width Test.txt** script.

1. Send the HitMap\_Deptime\_Wr command with a data argument of 0. Verify this command with the HitMap\_Deptime\_Rd command.
2. Send the GARC\_Mode\_Wr command with a data argument of decimal 768 to ensure the test pin multiplexer is set to the HitMap\_Test output. Place an oscilloscope probe on this test point.
3. Send the HitMap\_Width\_Wr command with a data argument of 0. Verify the command using the HitMap\_Width\_Rd readback.
4. Send the HitMap\_Delay\_Wr command with a data argument of 0. Verify the command using the HitMap\_Delay\_Rd readback.
5. Send the HitMap\_Deptime\_Wr command with a data argument of 0. Verify the command using the HitMap\_Deptime\_Rd readback.
6. Set up the test pulser to stimulate the discriminator input for channel 0. Monitor the HitMap Test output with the oscilloscope triggered from the test pulser. Using the HitMap\_Width\_Wr command with data arguments of 0 – 15, verify the HitMap Width varies as expected and record the data in the table below.

HitMap Width Command	Measured Width (ns)	Expected Width (ns)
0		150
1		200
2		250
3		300
4		350
5		400

6		450
7		500
8		550
9		600
10		650
11		700
12		750
13		800
14		850
15		900

7. Send the HitMap\_Width\_Wr command with a data argument of 0 to reset the HitMap Width. Verify the command with the HitMap\_Width\_Rd command.

## 12.2 HitMap Delay Test

This setup is the same as the previous section with the exception that the oscilloscope is in persistence mode and the trigger is on the discriminator input. The minimum delay is the number to be recorded. This test may be automated using the LabView GSE via the GARC HitMap Delay Test.txt script.

1. Using the HitMap\_Delay\_Wr command with data arguments of 0 – 31, verify the HitMap Delay varies as expected and record the data in the table below.

HitMap Delay Command	Measured Delay (ns)	Expected Delay (ns)
0		900
1		950
2		1000
3		1050
4		1100
5		1150
6		1200
7		1250
8		1300
9		1350
10		1400
11		1450
12		1500
13		1550
14		1600
15		1650
16		1700
17		1750
18		1800
19		1850
20		1900
21		1950
22		2000
23		2050

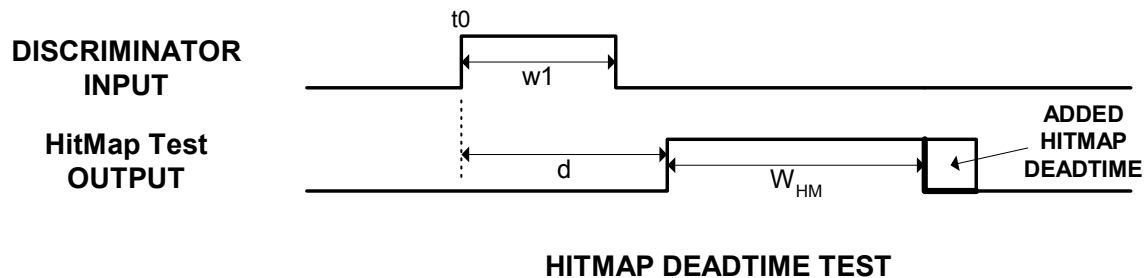
24		2100
25		2150
26		2200
27		2250
28		2300
29		2350
30		2400
31		2450

2. Send the HitMap\_Delay\_Wr command with a data argument of 0 to reset the HitMap Delay. Verify the command with the HitMap\_Delay\_Rd command.

### 12.3 HitMap Deadtime Stretch Test

This test may be automated using the LabView GSE via the GARC HitMap Deadtime Test.txt script.

This test will monitor the function of the HitMap\_Deadtime adjustment. This Deadtime register adds a time (0 to 350 ns) to the end of the HitMap pulse, starting at the end of the HitMap Width calculation. The diagram below illustrates the position of the added time window.



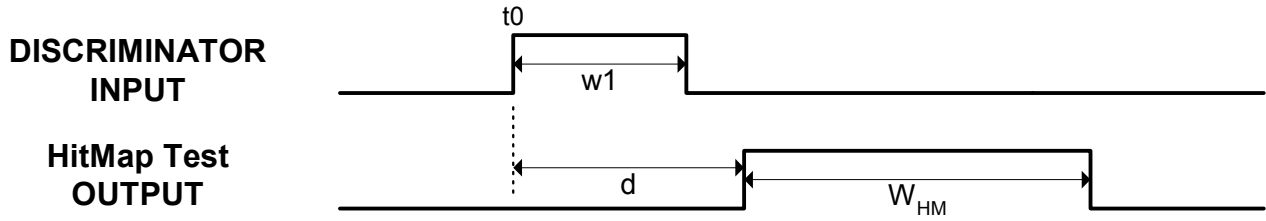
1. Send the HitMap\_Deadtime\_Wr command with a data argument of 0. Verify this command with the HitMap\_Deadtime\_Rd command.
2. Using the HitMap\_Deadtime\_Wr command with data arguments of 0 – 7, verify the HitMap Deadtime extends the trailing edge of the HitMap test pulse as expected and record the data in the table below.

HitMap Deadtime Command	Measured Deadtime Pulse Extension - (ns)	Expected Deadtime Pulse Extension -(ns)
0		0
1		50
2		100
3		150
4		200
5		250
6		300
7		350

3. Send the GARC\_Reset command to restore all register values to their initial state.

### 12.4 HitMap Minimum Width Test

This test characterizes the performance of the HitMap functionality for a minimum width discriminator input.

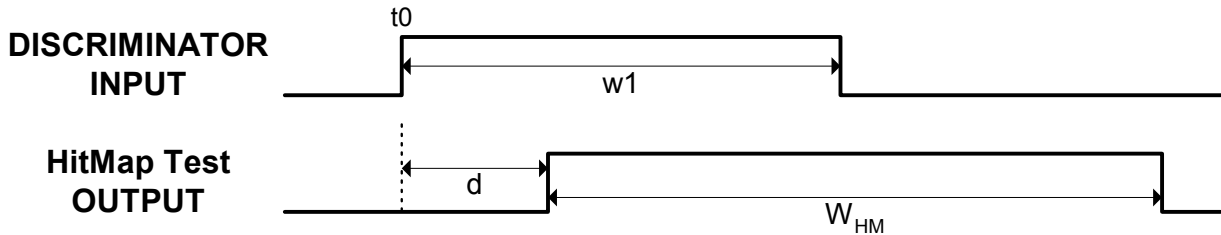


#### HITMAP MINIMUM WIDTH TEST

1. Send the HitMap\_Delay\_Wr command with a data argument of 0. Verify the command with the HitMap\_Delay\_Rd command. This fixes the delay ( $d$  in the figure) at 850 ns for this test.
2. Send the HitMap\_Width\_Wr command with a data argument of 0. Verify the command with the HitMap\_Width\_Rd command. This fixes the output width ( $W_{HM}$  in the figure) at 150 ns for this test.
3. Starting with the discriminator input width at 200 ns, note that the HitMap output is shown as above. Decrease the width of the discriminator input until the HitMap output starts to disappear. Verify that the HitMap\_Test is stable for all discriminator inputs greater than 50 ns.

### 12.5 HitMap Extended Pulse Test

This test characterizes the performance of the HitMap circuitry with an extended discriminator input pulse.

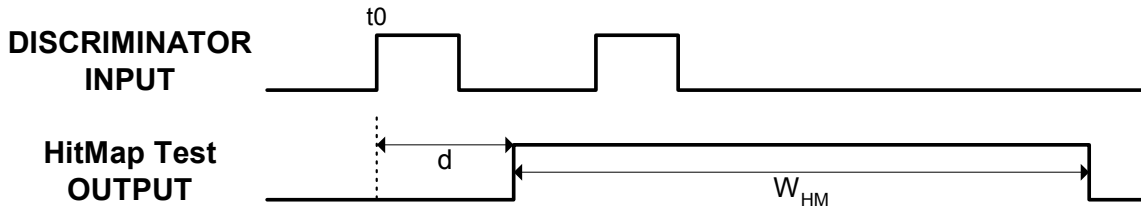


#### HITMAP EXTENDED PULSE TEST

1. Send the HitMap\_Delay\_Wr command with a data argument of 0. Verify the command with the HitMap\_Delay\_Rd command. This fixes the delay ( $d$  in the figure) at 850 ns for this test.
2. Send the HitMap\_Width\_Wr command with a data argument of 0. Verify the command with the HitMap\_Width\_Rd command. This fixes the output width ( $W_{HM}$  in the figure) at 150 ns for this test.
3. Starting with the discriminator input width at about 500 ns, increase the discriminator width up to approximately 10  $\mu$ sec. Verify that the HitMap\_Width increases with the discriminator input width.

### 12.6 HitMap Double Pulse Test

This test characterizes the performance of the HitMap circuitry with a double pulse at the discriminator input. The discriminator input width should be set to be approximately 100 ns.



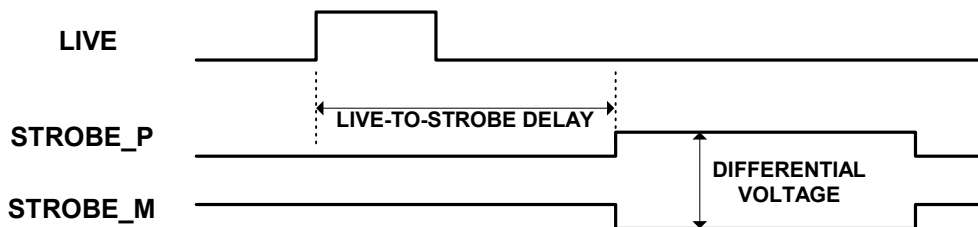
#### HITMAP DOUBLE PULSE TEST

1. Send the HitMap\_Delay\_Wr command with a data argument of 0. Verify the command with the HitMap\_Delay\_Rd command. This fixes the delay (**d** in the figure) at 900 ns for this test.
2. Send the HitMap\_Width\_Wr command with a data argument of 0. Verify the command with the HitMap\_Width\_Rd command. This fixes the output width (**W<sub>HM</sub>** in the figure) at 150 ns for this test.
3. Send the HitMap\_Deadtime\_Wr command with a data argument of 0. Verify the command with the HitMap\_Deadtime\_Rd command.
4. Starting with the discriminator input pulses at about 100 ns in width and 10  $\mu$ s apart, note that there are two HitMap output pulses each 150 ns wide.
5. Decrease the distance between the discriminator input pulses until the two HitMap output pulses merge to one extended pulse. Verify that this merge time is approximately  $150+900+50 = 1100$  ns. Verify that the length of the pulse is extended 950 ns from the leading edge of the second pulse.

Measured Merge Time - \_\_\_\_\_ (Expected Value 1100ns)

### 13.0 Strobe Test

This section tests the proper functioning of the commandable strobe pulse to the GAFEs from the GARC. This signal is used as a level command to the GAFE(s) to indicate the timing of the test charge injection to the amplifier front end. Using the oscilloscope monitoring Live on GARC pin 179, GAFE\_STROBEP on GARC pin 163 and GAFE\_STROBEM on GARC pin 164, perform the following (as shown in the diagram below):



#### STROBE COMMAND TEST

This test will require the LabView GSE to be running in the Internal Pulser mode. The GARC Live signal may be monitored on J7-8 on the GARC test board. STROBE\_P is at J4-10 and STROBE\_M is at J4-9 on the GARC test board.

1. Send the ADC\_TACQ\_Wr command with a variable of 0. Verify command with a ADC\_TACQ\_Rd command.
2. Measure the DC levels of both STROBE\_P and STROBE\_M and record the values below
  - a. **STROBE\_P (DC):** \_\_\_\_\_ (expected ~ 700 mV)
  - b. **STROBE\_M (DC):** \_\_\_\_\_ (expected ~1600 mV)
3. Send the GARC\_Mode\_Wr command with data argument 1792 to switch the test pin multiplexer to the Live signal.
4. Set the scope to acquire on the posedge of Live. Send the GARC\_Cal\_Strobe command (this is a dataless command so the data argument is 0).
5. Verify the STROBE command has executed by noting the differential signal on the two oscilloscope channels. It is expected that the voltage amplitude of each side of the differential is greater than 50 mV and that the pulse duration is approximately 10  $\mu$ sec.

**STROBEP to STROBEM differential voltage:** \_\_\_\_\_ (expected 900 mV)

**Live-to-STROBE delay:** \_\_\_\_\_ (expected ~ 150 ns)

**STROBE pulse duration ( $\mu$ sec):** \_\_\_\_\_ (expected 12.5  $\mu$ sec)

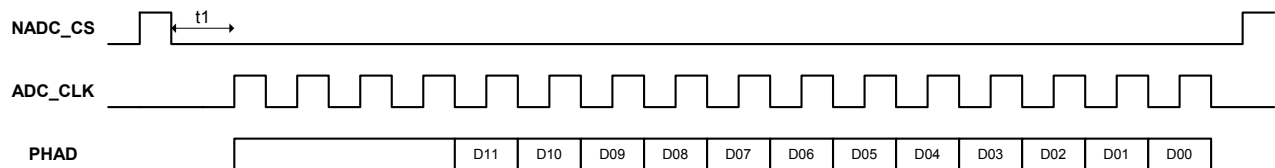
6. Send the GARC\_Reset command to restore all register values to their initial state.

#### 14.0 Capture of the ADC Control Signals

1. Connect the probe on channel 1 to NADC\_CS (J4-1). Set the scope to trigger on the negedge of this signal.
2. Connect the probe on channel 2 to ADC\_CLK (J4-2).
3. Connect the probe on channel 3 to PHAD00 (J4-15).
4. Send the TRIG\_ZS command to initiate an analog-to-digital conversion. The scope should trigger and the following waveform should be displayed. Verify that the ADC clock is nominally 2 MHz.

**Clock rate measured:** \_\_\_\_\_ MHz

5. The time t1 represents the total ADC TACQ (ADC acquisition time).

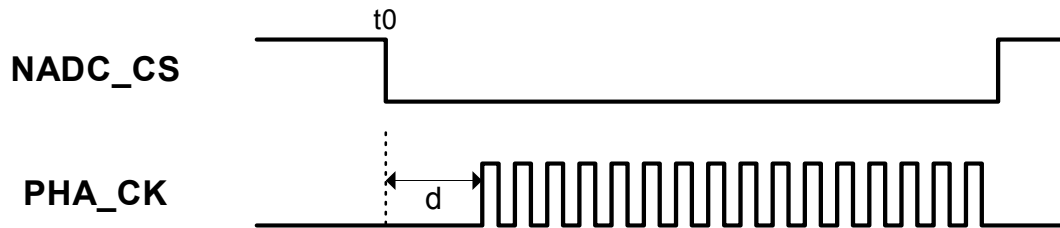


MAX145 ADC CONVERSION CYCLE

6. Measure the duration of the NADC\_CS: \_\_\_\_\_ (expected ~ 10.6  $\mu$ sec)

### 15.0 ADC TACQ Test

1. The next parameter to be measured will be the ADC time to acquisition (essentially the settling time allowed for the analog signal). This is a command with a 6 bit data argument. This alters the time between the ADC chip select transitioning active low and the start of the ADC clock. The register is set with the ADC\_TACQ\_Wr command and verified with the ADC\_TACQ\_Rd command. For each value of ADC\_TACQ in the table below, record the time from the falling edge of the ADC\_CS\_N and the rising edge of the first ADC clock. Nominally, with an ADC\_TACQ register setting of 0, there will be a CS → PHA clock delay time of 2850 ns. There is the potential for a small amount of synchronization jitter (~few clocks) within the state machine on this parameter.



#### ADC\_TACQ DELAY MEASUREMENT

NADC\_CS is at J4-1 on the GARC test board. PHA\_CK is at J4-2.

This test may be automated using the LabView GSE script GARC\_ADC\_TACQ\_Test.txt.

For each ACD\_TACQ step, measure the delay, **d**, as shown in the diagram above. For the table below, the important parameter is the even 50 ns spacing on each step from the original zero point.

ADC TACQ Register	Measured Time CS → PHA Clock (ns)	Expected Time CS → PHA Clock (ns)
0		2850
1		2900
2		2950
3		3000
4		3050
5		3100
6		3150
7		3200
8		3250
9		3300
10		3350
11		3400
12		3450
13		3500
14		3550
15		3600
16		3650
17		3700
18		3750
19		3800
20		3850
21		3900
22		3950
23		4000

24		4050
25		4100
26		4150
27		4200
28		4250
29		4300
30		4350
31		4400
32		4450
33		4500
34		4550
35		4600
36		4650
37		4700
38		4750
39		4800
40		4850
41		4900
42		4950
43		5000
44		5050
45		5100
46		5150
47		5200
48		5250
49		5300
50		5350
51		5400
52		5450
53		5500
54		5550
55		5600
56		5650
57		5700
58		5750
59		5800
60		5850
61		5900
62		5950
63		6000

2. Send the GARC\_Reset command to restore all register values to their initial state.

## **16.0 Test of the GARC LVDS Circuitry Driver Currents**

This section tests the proper functioning of the GARC LVDS Driver circuitry. It is required that all LVDS drivers be terminated at the receiver with a 100 ohm resistor. The ACD-to-AEM nominal drive current is 3.5 mA across the 100 ohms for a voltage differential of 350 mV.

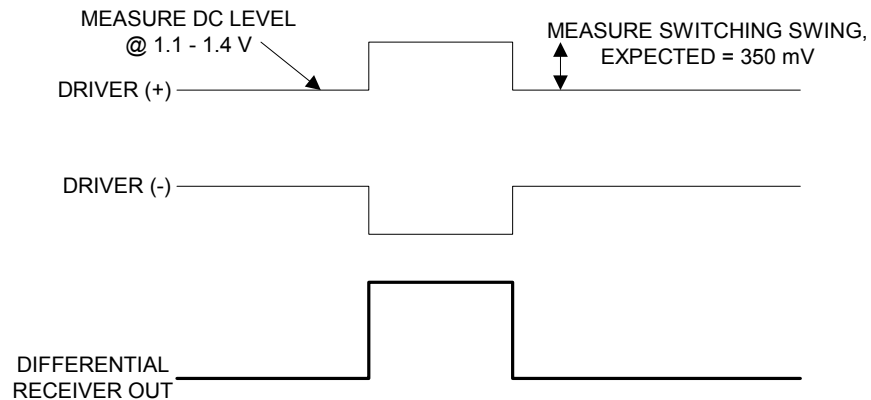
There are 6 LVDS receiver input pairs to the GARC:

- a) ACD\_CLK\_A, ACD\_CLK\_B
- b) ACD\_CMD\_A, ACD\_CMD\_B
- c) ACD\_RESET\_A, ACD\_RESET\_B

There are 40 LVDS driver output pairs from the GARC:

- d) ACD\_NSDATA\_A, ACD\_NSDATA\_B
- e) ACD\_NVETO\_nnA, ACD\_NVETO\_nnB (where nn is 00 – 17)
- f) ACD\_CNO\_A, ACD\_CNO\_B

The “A” side interface will be examined first. Verify that an AEM (or AEM simulator) is connected to the GARC “A” side interface. Using a digitizing oscilloscope, measure the DC baseline and switching differential voltage on each of the driver/receiver pairs. Two scope probes may be used, one on each side of the differential receiver inputs. The DC level and switching voltages may be measured as diagrammed below. The measurement will be made with the Drivers enabled and with them disabled.



Record the information for each pin on the Primary side interface in the table below. Please document the data from the following steps in the copy of the Test Record associated with this procedure.

Obtain a scope picture of the nominal pulse Amplitude Swings

GARC Signal	Signal Location	- DC Level (V) Enabled	+ DC Level (V) Enabled	- DC Level (V) Disabled	+ DC Level (V) Disabled
ACD_NVETO_12B	R1				
ACD_NVETO_12A	R2				
ACD_NVETO_11B	R3				
ACD_NVETO_11A	R4				
ACD_NVETO_10B	R5				
ACD_NVETO_10A	R6				
ACD_NVETO_09B	R7				
ACD_NVETO_09A	R8				
ACD_NVETO_08B	R9				
ACD_NVETO_08A	R10				
ACD_NVETO_07B	R11				
ACD_NVETO_07A	R12				
ACD_NVETO_06B	R13				
ACD_NVETO_06A	R14				
ACD_NVETO_05B	R15				
ACD_NVETO_05A	R16				
ACD_NVETO_04B	R17				
ACD_NVETO_04A	R18				
ACD_NVETO_03B	R19				
ACD_NVETO_03A	R20				
ACD_NVETO_02B	R21				
ACD_NVETO_02A	R22				

ACD NVETO 01B	R23				
ACD NVETO 01A	R24				
ACD NVETO 00B	R25				
ACD NVETO 00A	R26				
ACD CNO B	R27				
ACD CNO A	R28				
ACD NVETO 13B	R31				
ACD NVETO 13A	R32				
ACD NVETO 14B	R33				
ACD NVETO 14A	R34				
ACD NVETO 15B	R35				
ACD NVETO 15A	R36				
ACD NVETO 16B	R37				
ACD NVETO 16A	R38				
ACD NVETO 17B	R39				
ACD NVETO 17A	R40				
ACD NSDATA B	R29				
ACD NSDATA A	R30				

Verify the signal at J3-3 is +3.3V and 1uSec wide. Gnd is on J3-6.

Verified - \_\_\_\_\_

Measure the GAFE to GARC LVDS signals.

GARC Signal	Test Board Pin	DC Level (V)	Switch Swing (V)
IRTN 00	J4-33		
DISC 00	J4-34		
CHID 00	J4-35		
IRTN 01	J4-36		
DISC 01	J4-37		
CHID 01	J4-38		
IRTN 02	J4-39		
DISC 02	J4-40		
CHID 02	J4-41		
IRTN 03	J4-42		
DISC 03	J4-43		
CHID 03	J4-44		
IRTN 04	J4-45		
DISC 04	J4-46		
CHID 04	J4-47		
IRTN 05	J4-48		
DISC 05	J4-49		
CHID 05	J4-50		
IRTN 06	J4-51		
DISC 06	J4-52		
CHID 06	J4-53		
IRTN 07	J4-54		
DISC 07	J4-55		
CHID 07	J4-56		
IRTN 08	J4-57		
DISC 08	J4-58		
CHID 08	J4-59		
IRTN 09	J4-60		
DISC 09	J4-61		

CHID 09	J4-62		
IRTN 10	J4-63		
DISC 10	J4-64		
CHID 10	J4-65		
IRTN 11	J4-66		
DISC 11	J4-67		
CHID 11	J4-68		
IRTN 12	J4-69		
DISC 12	J4-70		
CHID 12	J4-71		
IRTN 13	J4-72		
DISC 13	J4-73		
CHID 13	J4-74		
IRTN 14	J4-75		
DISC 14	J4-76		
CHID 14	J4-77		
IRTN 15	J4-78		
DISC 15	J4-79		
CHID 15	J4-80		
IRTN 16	J4-81		
DISC 16	J4-82		
CHID 16	J4-83		
IRTN 17	J4-84		
DISC 17	J4-85		
CHID 17	J4-86		

This completes the testing of the GARC LVDS driver circuitry.

#### **17.0 Multiple System Clock Speed Test and Power Supply Rail Tests – 3.0V to 3.6V**

The function of this test is to verify operation of the GARC circuitry over the range of allowable clock frequencies and Power Supply Rail limits. The majority of this test has been performed at the nominal 20 MHz clock and has therefore verified the nominal frequency. This portion of the test will verify operation at the low and high limit frequencies and at min and max rail voltages. This will be verified by performing the register test. This may be done via automated script.

- 1) Set the clock frequency to the 22 MHz high limit.
- 2) Run the register test from section 6
- 3) Verify that all parameters are essentially unchanged

Register Test Verified - \_\_\_\_\_

- 4) Set power supply to +3.6V high limit.
- 5) Record the +3.6V Current

+3.6V current - \_\_\_\_\_

- 6) Run the register test from section 6
- 7) Verify that all parameters are essentially unchanged

Register Test Verified - \_\_\_\_\_

- 8) Set power supply to +3.0V high limit.
- 9) Record the +3.0V Current

+3.0V current - \_\_\_\_\_

- 10) Run the register test from section 6
- 11) Verify that all parameters are essentially unchanged

Register Test Verified - \_\_\_\_\_

Set the clock frequency to the 14 MHz low limit and reset the power supply to +3.3V.

- 12) Run the register test from section 6
- 13) Verify that all parameters are essentially unchanged

Register Test Verified - \_\_\_\_\_

- 14) Set power supply to +3.6V high limit.
- 15) Run the register test from section 6
- 16) Verify that all parameters are essentially unchanged

Register Test Verified - \_\_\_\_\_

- 17) Set power supply to +3.0V high limit.
- 18) Run the register test from section 6
- 19) Verify that all parameters are essentially unchanged

Register Test Verified - \_\_\_\_\_

## Appendix 1: GARC Documentation

A more complete set of documentation (such as the Verilog, EDIF, layout, and wirebonding diagram) is available on the LHEA ACD electronics web page at:

<http://lhea-glast.gsfc.nasa.gov/acd/electronics/>

## Appendix 2: GARC Configuration Command Format

The GARC logic core responds only to properly structured commands. There are two possible command types – Trigger Commands and Configuration Commands. Trigger commands initiate an Event Data cycle, causing a GAFE Hold, an analog-to-digital conversion, and the return of an event data packet. Configuration commands are used to either read or write GARC or GAFE registers.

The format for Trigger Commands is detailed in the table below.

Bit(s)	Bit Description	Value
3	Start Bit	1
2:1	Trigger Type Bits	10 for ZS Enabled Trigger 01 for Send All PHA Trigger
0	Parity Bit	Odd parity bit over previous two bits

Therefore, a 1100 is a ZS Enabled Trigger command and a 1010 is a Send All PHA Trigger command.

This format for GARC Configuration Commands is detailed in the table below.

Field	# bits	Function
Start	1	1 for start
CMD Type	2	00 for command
CMD Type Parity	1	Odd Parity over previous 2 bits (without Start bit)
GAFE/GARC Select	1	0 for GARC 1 for GAFE
GAFE/GARC Address	5	GAFE: Select which GAFE, 0x1F for all GAFE GARC: Select which function block
Read/Write	1	0 for write, 1 for read
Data/Dataless	1	0 for dataless, 1 for data, always 1 for ACD
register/function number	4	Which register/function in the function block
CMD Parity	1	Odd parity bit over previous 15 bits
Data	16	Data Field
Data Parity	1	Odd parity bit over previous 16 bits

**Appendix 3: GARC Event Data Format:**

The GARC core returns an event data packet when a valid trigger command is received. The event data format is detailed in the table below.

<u>Field</u>	<u># bits</u>	<u>Function</u>
Start Bit	1	1 for start
Hit Map Bits	18	Bits 17-0 for channels 0-17, bit set if hit in channel
Zero Suppression Bits	18	Bits 17-0 for channels 0-17, bit set if PHA above threshold
CMD/Data ERROR	1	Error in command parity detected
Header Parity	1	Odd parity bit over previous 37 bits
PHA Words (quantity 0-18) Order: Channel 0 to channel 17	15	Bit 14: 1 if another PHA word follows this one, 0 if this is last one Bit 13: 1 for high range, 0 for low range Bits 12-1: the PHA value, 0 to 4095 Bit 0: Odd parity over last 14 bits

**Appendix 4: GARC Configuration Data Readback Format:**

The GARC core returns a register configuration data packet when a valid configuration readback command is received. The configuration readback data format is detailed in the table below.

<u>Field</u>	<u># bits</u>	<u>Function</u>
Start	1	1 for start
GAFE/GARC Select	1	Copy of write command field (0 for GARC, 1 for GAFE)
GAFE/GARC Address	5	Copy of write command field (Select which GAFE)
Read/Write	1	1 for read
Data/Dataless	1	always 1
Register/function number	4	Copy of write command field (which register/function in the function block)
CMD Parity	1	Odd parity bit over previous 12 bits
Data	16	Data, MSB first
CMD/DATA ERROR	1	Error in parity detected
Parity	1	Odd parity bit over previous 17 bits

**Appendix 5: GARC Pin List**

<b>Tanner IO Cell</b>	<b>MOSIS Bond Pad Name</b>	<b>GARC Package Pin Number</b>	<b>GARC Signal Name</b>
PADGnd	L1	1	DGND
PADAREF	L2	2	ACD_CLK_AP
PADAREF	L3	3	ACD_CLK_AM
PADAREF	L4	4	ACD_CLK_BP
PADAREF	L5	5	ACD_CLK_BM
PADAREF	L6	6	ACD_NSCMD_AP
PADAREF	L7	7	ACD_NSCMD_AM
PADAREF	K8	8	ACD_NSCMD_BP
PADAREF	L9	9	ACD_NSCMD_BM
PADAREF	L10	10	ACD_NRST_AP
PADAREF	L11	11	ACD_NRST_AM
PADAREF	L12	12	ACD_NRST_BP

PADAREF	L13	13	ACD_NRST_BM
PADAREF	L14	14	ACD_NSDATA_AP
PADAREF	L15	15	ACD_NSDATA_AM
PADAREF	L16	16	ACD_NSDATA_BP
PADAREF	L17	17	ACD_NSDATA_BM
PADAREF	L18	18	ACD_NCNO_AP
PADAREF	L19	19	ACD_NCNO_AM
PADAREF	L20	20	ACD_NCNO_BP
PADAREF	L21	21	ACD_NCNO_BM
PADAREF	L22	22	ACD_NVETO_00AP
PADAREF	L23	23	ACD_NVETO_00AM
PADVdd	L24	24	DVDD
PADAREF	L25	25	ACD_NVETO_00BP
PADAREF	L26	26	ACD_NVETO_00BM
PADGnd	L27	27	DGND
PADAREF	L28	28	ACD_NVETO_01AP
PADAREF	L29	29	ACD_NVETO_01AM
PADAREF	L30	30	ACD_NVETO_01BP
PADAREF	L31	31	ACD_NVETO_01BM
PADAREF	L32	32	ACD_NVETO_02AP
PADAREF	L33	33	ACD_NVETO_02AM
PADAREF	L34	34	ACD_NVETO_02BP
PADAREF	L35	35	ACD_NVETO_02BM
PADAREF	L36	36	ACD_NVETO_03AP
PADAREF	L37	37	ACD_NVETO_03AM
PADAREF	L38	38	ACD_NVETO_03BP
PADAREF	L39	39	ACD_NVETO_03BM
PADAREF	L40	40	ACD_NVETO_04AP
PADAREF	L41	41	ACD_NVETO_04AM
PADAREF	L42	42	ACD_NVETO_04BP
PADAREF	L43	43	ACD_NVETO_04BM
PADAREF	L44	44	ACD_NVETO_05AP
PADAREF	L45	45	ACD_NVETO_05AM
PADAREF	L46	46	ACD_NVETO_05BP
PADAREF	L47	47	ACD_NVETO_05BM
PADAREF	L48	48	ACD_NVETO_06AP
PADAREF	L49	49	ACD_NVETO_06AM
PADAREF	L50	50	ACD_NVETO_06BP
PADAREF	L51	51	ACD_NVETO_06BM
PADVdd	L52	52	DVDD
PADGnd	B1	53	DGND
PADAREF	B2	54	ACD_NVETO_07AP
PADAREF	B3	55	ACD_NVETO_07AM
PADAREF	B4	56	ACD_NVETO_07BP
PADAREF	B5	57	ACD_NVETO_07BM
PADAREF	B6	58	ACD_NVETO_08AP
PADAREF	B7	59	ACD_NVETO_08AM
PADAREF	B8	60	ACD_NVETO_08BP
PADAREF	B9	61	ACD_NVETO_08BM
PADAREF	B10	62	ACD_NVETO_09AP
PADAREF	B11	63	ACD_NVETO_09AM
PADAREF	B12	64	ACD_NVETO_09BP
PADAREF	B13	65	ACD_NVETO_09BM
PADAREF	B14	66	ACD_NVETO_10AP
PADAREF	B15	67	ACD_NVETO_10AM
PADAREF	B16	68	ACD_NVETO_10BP
PADAREF	B17	69	ACD_NVETO_10BM
PADAREF	B18	70	ACD_NVETO_11AP
PADAREF	B19	71	ACD_NVETO_11AM
PADAREF	B20	72	ACD_NVETO_11BP
PADAREF	B21	73	ACD_NVETO_11BM

PADAREF	B22	74	ACD_NVETO_12AP
PADAREF	B23	75	ACD_NVETO_12AM
PADAREF	B24	76	ACD_NVETO_12BP
PADAREF	B25	77	ACD_NVETO_12BM
PADVdd	B26	78	DVDD
PADAREF	B27	79	CHID_17
PADGnd	B28	80	DGND
PADAREF	B29	81	DISC_17
PADAREF	B30	82	IRTN_17
PADAREF	B31	83	CHID_16
PADAREF	B32	84	DISC_16
PADAREF	B33	85	IRTN_16
PADAREF	B34	86	CHID_15
PADAREF	B35	87	DISC_15
PADAREF	B36	88	IRTN_15
PADAREF	B37	89	CHID_14
PADAREF	B38	90	DISC_14
PADAREF	B39	91	IRTN_14
PADAREF	B40	92	CHID_13
PADAREF	B41	93	DISC_13
PADAREF	B42	94	IRTN_13
PADAREF	B43	95	CHID_12
PADAREF	B44	96	DISC_12
PADAREF	B45	97	IRTN_12
PADAREF	B46	98	CHID_11
PADAREF	B47	99	DISC_11
PADAREF	B48	100	IRTN_11
PADAREF	B49	101	CHID_10
PADAREF	B50	102	DISC_10
PADAREF	B51	103	IRTN_10
PADAREF	B52	104	HLD_WOR_BIAS
PADAREF	R52	105	CHID_09
PADAREF	R51	106	DISC_09
PADAREF	R50	107	IRTN_09
PADAREF	R49	108	CHID_08
PADAREF	R48	109	DISC_08
PADAREF	R47	110	IRTN_08
PADAREF	R46	111	CHID_07
PADAREF	R45	112	DISC_07
PADAREF	R44	113	IRTN_07
PADAREF	R43	114	CHID_06
PADAREF	R42	115	DISC_06
PADAREF	R41	116	IRTN_06
PADAREF	R40	117	CHID_05
PADAREF	R39	118	DISC_05
PADAREF	R38	119	IRTN_05
PADAREF	R37	120	CHID_04
PADAREF	R36	121	DISC_04
PADAREF	R35	122	IRTN_04
PADAREF	R34	123	CHID_03
PADAREF	R33	124	DISC_03
PADAREF	R32	125	IRTN_03
PADAREF	R31	126	CHID_02
PADAREF	R30	127	DISC_02
PADAREF	R29	128	IRTN_02
PADAREF	R28	129	CHID_01
PADAREF	R27	130	DISC_01
PADAREF	R26	131	IRTN_01
PADVdd	R25	132	DVDD
PADAREF	R24	133	CHID_00
PADAREF	R23	134	DISC_00

PADAREF	R22	135	IRTN_00
PADGnd	R21	136	DGND
PADInC	R20	137	PHAD_17
PADInC	R19	138	PHAD_16
PADInC	R18	139	PHAD_15
PADInC	R17	140	PHAD_14
PADInC	R16	141	PHAD_13
PADInC	R15	142	PHAD_12
PADInC	R14	143	PHAD_11
PADInC	R13	144	PHAD_10
PADInC	R12	145	PHAD_09
PADInC	R11	146	PHAD_08
PADInC	R10	147	PHAD_07
PADInC	R9	148	PHAD_06
PADInC	R8	149	PHAD_05
PADInC	R7	150	PHAD_04
PADInC	R6	151	PHAD_03
PADInC	R5	152	PHAD_02
PADInC	R4	153	PHAD_01
PADInC	R3	154	PHAD_00
PADVdd	R2	155	DVDD
PADAREF	R1	156	BIAS_RCVR
PADGnd	T52	157	DGND
PADAREF	T51	158	IRTN_HL_DISC
PADAREF	T50	159	OR_HL_DISC
PADAREF	T49	160	BIAS_DRV_H
PADAREF	T48	161	GAFE_HOLDP
PADAREF	T47	162	GAFE_HOLDM
PADAREF	T46	163	GAFE_STROBEP
PADAREF	T45	164	GAFE_STROBEM
PADAREF	T44	165	GAFE_DATP
PADAREF	T43	166	GAFE_DATM
PADAREF	T42	167	GAFE_CLKP
PADAREF	T41	168	GAFE_CLKM
PADAREF	T40	169	BIAS_DRV_L
PADInC	T39	170	GAFE_RET_DATA
PadOut	T38	171	GAFE_RST
PADOut	T37	172	ADC_CLK
PADOut	T36	173	NADC_CS
PADOut	T35	174	DAC_CLK
PADOut	T34	175	DAC_DATA
PADInC	T33	176	DAC_READBACK
PADOut	T32	177	NDAC_CS
PADOut	T31	178	NDAC_CLR
PADOut	T30	179	HITMAP_TEST
PADOut	T29	180	TRIG_TST
PADInC	T28	181	FREE_ID
PADInC	T27	182	PWR_ON_RST
PADOut	T26	183	VETO_ENA_TST
PADAREF	T25	184	LVDS_PRESETADJ
PADOut	T24	185	HV_ENABLE_1
PADOut	T23	186	HV_ENABLE_2
PADGnd	T22	187	DGND
PADAREF	T21	188	ACD_NVETO_17AP
PADAREF	T20	189	ACD_NVETO_17AM
PADAREF	T19	190	ACD_NVETO_17BP
PADAREF	T18	191	ACD_NVETO_17BM
PADAREF	T17	192	ACD_NVETO_16AP
PADAREF	T16	193	ACD_NVETO_16AM
PADAREF	T15	194	ACD_NVETO_16BP
PADAREF	T14	195	ACD_NVETO_16BM

PADAREF	T13	196	ACD_NVETO_15AP
PADAREF	T12	197	ACD_NVETO_15AM
PADAREF	T11	198	ACD_NVETO_15BP
PADAREF	T10	199	ACD_NVETO_15BM
PADAREF	T9	200	ACD_NVETO_14AP
PADAREF	T8	201	ACD_NVETO_14AM
PADAREF	T7	202	ACD_NVETO_14BP
PADAREF	T6	203	ACD_NVETO_14BM
PADAREF	T5	204	ACD_NVETO_13AP
PADAREF	T4	205	ACD_NVETO_13AM
PADAREF	T3	206	ACD_NVETO_13BP
PADAREF	T2	207	ACD_NVETO_13BM
PADVdd	T1	208	DVDD

### Appendix 6: Test Results Record

All measurements and test results will be recorded in this Test Results Record. A copy of the entire this entire document is not required each time the test is performed. The appendix will serve as the official record each time this test is performed.

#### 2.0 QA Signoff

Quality Assurance approval to proceed with the test - \_\_\_\_\_

(NOTE: If QA gives verbal approval to proceed but is not in attendance for the test, record the date and the time of approval.)

BIAS Resistor Verified on the GARC Test Board - \_\_\_\_\_

#### 2.1 GARC Identification

The test conductor for this test is: \_\_\_\_\_

Date of test is: \_\_\_\_\_

The identification/Serial Number listed on the GARC ASIC is: \_\_\_\_\_

The serial number of the AEM Simulator board is: \_\_\_\_\_

The serial number of the GARC Test board is: \_\_\_\_\_

The serial number of the GAFE Simulator board is: \_\_\_\_\_

#### 2.2 Test Equipment Utilized

Instrument Type	Manufacturer & Model Number	NASA ID Number	Calibration Due Date
Power Supply +3.3V			
Power Supply +5.0V			
Multimeter 1			
Multimeter 2			
Pulse Generator			
Oscilloscope			

#### 4.1 Measurement of the GARC Bias Resistors and Voltages

+3.3V Current - \_\_\_\_\_ (\_\_\_\_\_  $\pm$  10mA)

GARC Bias Signal	GARC Pin	Resistor Test Point	Expected Voltage	Measured Voltage
HLD_WOR_BIAS	104	R19	1.64V	
BIAS_RCVR	156	R16	1.10V	
BIAS_DRV_H	160	R15	1.53V	
BIAS_DRV_L	169	R13	1.75V	
LVDS_PRESET_ADJ	184	R20	1.52V	

#### 4.2 GARC Registers Initial Reset Test

Turn On reset verified - \_\_\_\_\_

Board Serial Number verified - \_\_\_\_\_

Reset Command verified - \_\_\_\_\_

#### 5.1 FREE Power Measurement at the Nominal Power Supply Voltage

GARC Mode	GARC_Mode_Wr Data Argument	+3.3V Current Measured (mA)	+3.3V Current Expected (mA)
LVDS "A" Drivers Enabled LVDS "B" Drivers Enabled	768		185 $\pm$ 10mA
LVDS "A" Drivers Enabled LVDS "B" Drivers Disabled	256		135 $\pm$ 10mA
LVDS "A" Drivers Disabled LVDS "B" Drivers Enabled	512		135 $\pm$ 10mA
LVDS "A" Drivers Disabled LVDS "B" Drivers Disabled	0		80 $\pm$ 10mA

#### 5.2 FREE Power Measurement at the Minimum Power Supply Voltage

GARC Mode	GARC_Mode_Wr Data Argument	+3.0V Current Measured (mA)	+3.0V Current Expected (mA)
LVDS "A" Drivers Enabled LVDS "B" Drivers Enabled	768		160 $\pm$ 10mA
LVDS "A" Drivers Enabled LVDS "B" Drivers Disabled	256		115 $\pm$ 10mA
LVDS "A" Drivers Disabled LVDS "B" Drivers Enabled	512		115 $\pm$ 10mA
LVDS "A" Drivers Disabled LVDS "B" Drivers Disabled	0		65 $\pm$ 10mA

**5.3 FREE Power Measurement at the Maximum Power Supply Voltage**

GARC Mode	GARC_Mode_Wr Data Argument	+3.6V Current Measured (mA)	+3.6V Current Expected (mA)
LVDS "A" Drivers Enabled LVDS "B" Drivers Enabled	768		215 ± 10mA
LVDS "A" Drivers Enabled LVDS "B" Drivers Disabled	256		155 ± 10mA
LVDS "A" Drivers Disabled LVDS "B" Drivers Enabled	512		155 ± 10mA
LVDS "A" Drivers Disabled LVDS "B" Drivers Disabled	0		95 ± 10mA

**6.0 GARC Register Read/Write Tests**

This test may be automated using the LabView GSE via the Register Test VI.

Test Completed - \_\_\_\_\_

**6.40 Test of the GARC Parity Test**

This test may be automated using the LabView GSE via the **GARC Parity Test.txt** script

Test Completed - \_\_\_\_\_

**6.41 Test of the GARC Diagnostic Register**

This test may be automated using the LabView GSE via the **GARC Diagnostic Status Reg Test.txt** script

Test Completed - \_\_\_\_\_

**6.42 Test of the GARC Command Counters**

This test may be automated using the LabView GSE via the **GARC Command Counter Test.txt** script

Test Completed - \_\_\_\_\_

**6.43 Test of the Look-At-Me Circuitry**

This test may be automated using the LabView GSE via the **GARC Look At Me Test.txt** script.

Test Completed - \_\_\_\_\_

**7.1 Maximum PHA Return Test**

This test may be automated using the LabView GSE via the **GARC\_Max\_PHA\_Return\_Test.txt** script.

Test Completed - \_\_\_\_\_

## 7.2 PHA Enable/Disable Test

This test may be automated using the LabView GSE via the GARC PHA Enable Test.txt script.

Test Completed - \_\_\_\_\_

## 7.3 PHA Threshold Verification Test

This test may be automated using the LabView GSE via the GARC Chip PHA Threshold Test.txt script.

Test Completed - \_\_\_\_\_

## 8.1 Capture of the DAC Control Signals

DAC Control Signals verified - \_\_\_\_\_

Record DAC Clock Frequency - \_\_\_\_\_ (5Mhz  $\pm$  \_\_\_\_\_)

Record Reset Pulse Width - \_\_\_\_\_ (150nSec  $\pm$  \_\_\_\_\_)

## 8.2 TEST OF THE SAA/HV NORMAL MODES

1. HVBS Level set to 2048 - \_\_\_\_\_
2. SAA Level set to 1024 - \_\_\_\_\_
3. Using a multimeter on the DAC voltage reference, verify that the MAX5121 reference is at 1.25V. This is J5-3 on the GARC test board.  
**Value observed:** \_\_\_\_\_ (1.25V  $\pm$  \_\_\_\_\_ mV)
4. Using a multimeter on the DAC output, verify that the MAX5121 output is at 0V. This is J5-1 on the GARC test board.  
**Value observed:** \_\_\_\_\_ (0V  $\pm$  \_\_\_\_\_ mV)
5. Send the Use\_HV\_Nominal command. Verify that the MAX5121 output goes to half scale, 0.625V at J5-1.  
**Value observed:** \_\_\_\_\_ (0.625V  $\pm$  \_\_\_\_\_ mV)
6. Send the DAC\_HVReg\_Rd command and verify the return data pattern from the MAX5121 DAC is decimal 10240.  
**Verified:** \_\_\_\_\_
7. Send the Use\_SAA\_Level command. Verify that the MAX5121 output goes to 1/4 scale, 0.312 V at J5-1.  
**Value observed:** \_\_\_\_\_ (0.325V  $\pm$  \_\_\_\_\_ mV)

8. Send the DAC\_SAAREg\_Rd command and verify the return data pattern from the MAX5121 DAC is decimal 9216.

Verified: \_\_\_\_\_

### 8.3 Test of the HVBS Triple Modular Redundancy Circuitry

This test may be automated using the LabView GSE via the GARC HV Enable Test.tst script.

Voltage at Pin J7-6 when Enabled - \_\_\_\_\_ (3.25V ± \_\_\_\_ mV)

Voltage at Pin J7-7 when Enabled - \_\_\_\_\_ (3.25V ± \_\_\_\_ mV)

Voltage at Pin J7-6 when Disabled - \_\_\_\_\_ (0.015V ± \_\_\_\_ mV)

Voltage at Pin J7-7 when Disabled - \_\_\_\_\_ (0.015V ± \_\_\_\_ mV)

Test Completed - \_\_\_\_\_

### 9.0 GARC Test Pin Mux Verification

Monitor J7-8

Test Completed - \_\_\_\_\_

### 10.0 Test of the Hold Delay Operation

Trigger on “TRIG TST” at J7-9 and monitor “Hold” at J4-12.

GAFE\_HOLD\_P Voltage (J4-12) - \_\_\_\_\_ (870mV ± \_\_\_\_ mV)

GAFE\_HOLD\_M Voltage (J4-11) - \_\_\_\_\_ (1600mV ± \_\_\_\_ mV)

The LabView test script GARC Hold Delay Test.txt may be used to partially automate this test.

Test Completed - \_\_\_\_\_

### 11.1 Veto Delay Test

This test may be automated using the LabView GSE via the script GARC VETO Delay Test.txt.

Test Completed - \_\_\_\_\_

### 11.2 Veto Width Test

The setup for this test is the same as the VETO Delay test above. This test may be automated using the LabView GSE via the GARC VETO Width Test.txt script

Test Completed - \_\_\_\_\_

### 11.3 Veto Enable and Disable Test

This test may be automated when using the LabView GSE via test script **GARC VETO Disable A/B Side Test.txt**. Monitor the counters to verify the Veto signals.

1. Send the GARC\_Mode\_Wr command with a value of 768 to enable the “A” and enable “B” side VETOs.

R1 - \_\_\_\_\_ (expected 400mV  $\pm$  \_\_\_\_ mV)

R2 - \_\_\_\_\_ (expected 400mV  $\pm$  \_\_\_\_ mV)

2. Send the GARC\_Mode\_Wr command with a value of 512 to disable the “A” and enable “B” side VETOs.

R1 - \_\_\_\_\_ (expected 400mV  $\pm$  \_\_\_\_ mV)

R2 - \_\_\_\_\_ (expected 50mV  $\pm$  \_\_\_\_ mV)

3. Send the GARC\_Mode\_Wr command with a value of 256 to enable the “A” and disable “B” side VETOs.

R1 - \_\_\_\_\_ (expected 50mV  $\pm$  \_\_\_\_ mV)

R2 - \_\_\_\_\_ (expected 400mV  $\pm$  \_\_\_\_ mV)

4. Send the GARC\_Mode\_Wr command with a value of 0 to disable both the “A” and “B” side VETOs.

R1 - \_\_\_\_\_ (expected 50mV  $\pm$  \_\_\_\_ mV)

R2 - \_\_\_\_\_ (expected 50mV  $\pm$  \_\_\_\_ mV)

Side A Test Completed - \_\_\_\_\_

Side B Test Completed - \_\_\_\_\_

### 11.4 Veto Minimum Width Test

Test Completed - \_\_\_\_\_

### 11.5 Veto Double Pulse Test

Test Completed - \_\_\_\_\_

### 11.6 Veto Minimum Width Test

Test Completed - \_\_\_\_\_

### 12.1 HitMap Width Test

This test may be automated using the LabView GSE via the **GARC HitMap Width Test.txt** script.

Test Completed - \_\_\_\_\_

**12.2 HitMap Delay Test**

Test Skipped - \_\_\_\_\_

This test may be automated using the LabView GSE via the GARC HitMap Delay Test.txt script

Test Completed - \_\_\_\_\_

**12.3 HitMap Deadtime Stretch Test**

This test may be automated using the LabView GSE via the GARC HitMap Deadtime Test.txt script.

Test Completed - \_\_\_\_\_

**12.4 HitMap Minimum Width Test**

Test Completed - \_\_\_\_\_

**12.5 HitMap Double Pulse Test**

Test Completed - \_\_\_\_\_

**12.6 HitMap Minimum Width Test**

Measured Merge Time - \_\_\_\_\_ (Expected Value 1100nSec  $\pm$  \_\_\_\_ nSec)

Test Completed - \_\_\_\_\_

**13.0 Strobe Test**

Measure the DC levels of both STROBE\_P and STROBE\_M and record the values below

STROBE\_P (DC) (J4-10) : \_\_\_\_\_ (expected  $\sim 700$  mV  $\pm$  \_\_\_\_ mV)

STROBE\_M (DC) (J4-9) : \_\_\_\_\_ (expected  $\sim 1600$  mV  $\pm$  \_\_\_\_ mV)

STROBEP to STROBEM differential voltage: \_\_\_\_\_ (expected 900 mV  $\pm$  \_\_\_\_ mV)

Live-to-STROBE delay: \_\_\_\_\_ (expected  $\sim 150$  nSec  $\pm$  \_\_\_\_ nSec)

STROBE pulse duration ( $\mu$ sec): \_\_\_\_\_ (expected 12.5  $\mu$ Sec  $\pm$  \_\_\_\_  $\mu$ Sec)

**14.0 Capture of the ADC Control Signals**

Clock rate measured: \_\_\_\_\_ (5MHz  $\pm$  \_\_\_\_ MHz)

Duration \_\_\_\_\_ (expected  $\sim 10.6$   $\mu$ Sec  $\pm$  \_\_\_\_  $\mu$ Sec)

**15.0 ADC TACQ Test**

This test may be automated using the LabView GSE script GARC\_ADC\_TACQ\_Test.txt

Test Completed - \_\_\_\_\_

**16.0 Test of the GARC LVDS Circuitry Driver Currents**

Obtain a scope picture of the nominal pulse Amplitude Swings

GARC Signal	Signal Location	- DC Level (V) Enabled	+ DC Level (V) Enabled	- DC Level (V) Disabled	+ DC Level (V) Disabled
ACD_NVETO_12B	R1				
ACD_NVETO_12A	R2				
ACD_NVETO_11B	R3				
ACD_NVETO_11A	R4				
ACD_NVETO_10B	R5				
ACD_NVETO_10A	R6				
ACD_NVETO_09B	R7				
ACD_NVETO_09A	R8				
ACD_NVETO_08B	R9				
ACD_NVETO_08A	R10				
ACD_NVETO_07B	R11				
ACD_NVETO_07A	R12				
ACD_NVETO_06B	R13				
ACD_NVETO_06A	R14				
ACD_NVETO_05B	R15				
ACD_NVETO_05A	R16				
ACD_NVETO_04B	R17				
ACD_NVETO_04A	R18				
ACD_NVETO_03B	R19				
ACD_NVETO_03A	R20				
ACD_NVETO_02B	R21				
ACD_NVETO_02A	R22				
ACD_NVETO_01B	R23				
ACD_NVETO_01A	R24				
ACD_NVETO_00B	R25				
ACD_NVETO_00A	R26				
ACD_CNO_B	R27				
ACD_CNO_A	R28				
ACD_NVETO_13B	R31				
ACD_NVETO_13A	R32				
ACD_NVETO_14B	R33				
ACD_NVETO_14A	R34				
ACD_NVETO_15B	R35				
ACD_NVETO_15A	R36				
ACD_NVETO_16B	R37				
ACD_NVETO_16A	R38				
ACD_NVETO_17B	R39				
ACD_NVETO_17A	R40				
ACD_NSDATA_B	R29				
ACD_NSDATA_A	R30				

Verify the signal at J3-3 is +3.3V and 1uSec wide. Gnd is on J3-6.

Verified - \_\_\_\_\_

Measure the GAFE to GARC LVDS signals.

GARC Signal	Test Board Pin	DC Level (V)	Switch Swing (V)
IRTN 00	J4-33		
DISC 00	J4-34		
CHID 00	J4-35		
IRTN 01	J4-36		
DISC 01	J4-37		
CHID 01	J4-38		
IRTN 02	J4-39		
DISC 02	J4-40		
CHID 02	J4-41		
IRTN 03	J4-42		
DISC 03	J4-43		
CHID 03	J4-44		
IRTN 04	J4-45		
DISC 04	J4-46		
CHID 04	J4-47		
IRTN 05	J4-48		
DISC 05	J4-49		
CHID 05	J4-50		
IRTN 06	J4-51		
DISC 06	J4-52		
CHID 06	J4-53		
IRTN 07	J4-54		
DISC 07	J4-55		
CHID 07	J4-56		
IRTN 08	J4-57		
DISC 08	J4-58		
CHID 08	J4-59		
IRTN 09	J4-60		
DISC 09	J4-61		
CHID 09	J4-62		
IRTN 10	J4-63		
DISC 10	J4-64		
CHID 10	J4-65		
IRTN 11	J4-66		
DISC 11	J4-67		
CHID 11	J4-68		
IRTN 12	J4-69		
DISC 12	J4-70		
CHID 12	J4-71		
IRTN 13	J4-72		
DISC 13	J4-73		
CHID 13	J4-74		
IRTN 14	J4-75		
DISC 14	J4-76		
CHID 14	J4-77		
IRTN 15	J4-78		
DISC 15	J4-79		
CHID 15	J4-80		

IRTN 16	J4-81		
DISC 16	J4-82		
CHID 16	J4-83		
IRTN 17	J4-84		
DISC 17	J4-85		
CHID 17	J4-86		

**17.0 Clock Frequency and Power Rail Tests**

1) 22MHz Clock operation verified at all rail voltages - \_\_\_\_\_

+3.0V Current - \_\_\_\_\_

+3.6V Current - \_\_\_\_\_

2) 14MHz Clock operation verified at all rail voltages- \_\_\_\_\_

**END OF GARC TEST PROCEDURE**